En este informe se redacta la memoria final del proyecto Sistemas de control de acceso a sistemas informáticos alternativos a passwords en concepto de Trabajo de Fin de Grado.

Sistemas de control de acceso alternativos a passwords

Memoria Final

Autor: Ernesto Castañón Delgado
Tutor: Vicente Martín Ayuso

Tabla de contenido

1. Introducción.

La finalidad de este proyecto es la de estudiar las alternativas empleadas hoy en día para asegurar y garantizar el correcto acceso a infraestructuras sensibles, evitando el uso de las clásicas passwords por los problemas que acarrean, detallados en el apartado *2.2. Autenticación mediante passwords y problemas en su uso*.

Se **analizarán** y **estudiarán** algunas de las **soluciones más factibles** para garantizar un correcto **acceso a sistemas sensibles**, destacando las ventajas y problemas de cada una de las soluciones. Con esto se quiere decir que **no** se tratará de **solucionar** los problemas de seguridad y gestión de **acceso** a la hora de acceder a las diversas **plataformas web**, como puedan ser **redes sociales** o **sistemas de pago y/o transacciones**. El estudio se va a limitar al caso dado en el centro de supercomputación CeSViMa, pudiéndose extender a cualquier infraestructura física, siempre y cuando se tenga en cuenta que para el acceso a la infraestructura se pueden necesitar medios o periféricos hardware que, obviamente, serán controlados por la empresa o sistema alojado en dicha infraestructura.

Tras dicho estudio, se procederá al **diseño de la** que se considere **mejor opción** en una instalación real.

El **estudio** del sistema se llevará a cabo, tal y como se ha mencionado, en el **centro de supercomputación CeSViMa** ubicado en el Campus Montegancedo.

2. Análisis de técnicas de acreditación de acceso

2.1. Prefacio

Cada día se puede ver cómo la gente va confiando más en Internet, y por lo tanto en los ordenadores, para realizar su día a día, desde el ocio, como pueden ser las redes sociales, juegos, y hasta realizar compras o consultas bancarias.

Hace años la gente era más reacia al uso de estos sistemas, más aún cuando se debía conceder acceso en las distintas plataformas a sus cuentas bancarias o tarjetas de crédito. La gente desconfiaba de que alguien pudiera acceder a este tipo de datos porque principalmente, era su dinero lo que estaba en juego.

Sin embargo, por otro lado fue creciendo durante años el número de usuarios que empleaban sistemas como foros y sobre todo redes sociales, donde también estaban **confiando** (a veces inconscientemente) otro tipo de información, **su información, sus datos.**

Hoy en día, la mayoría de la gente ha cedido a la comodidad de emplear ordenadores y sobre todo Internet, donde pueden acceder de forma remota y más sencilla a realizar labores y consultar información. Pero todos estos sistemas fueron (y siguen siendo) protegidos comúnmente por un control de **usuario/password**, donde el sistema, al cual se desea acceder, posee cierta información almacenada para relacionar y autenticar a cada uno de sus usuarios. Este sistema de control de acceso, a pesar de que pueda parecer seguro, incluso a veces por estar protegido por comunicaciones cifradas (protocolo SSL [7]), tiene **muchos problemas** que se detallarán en el apartado *2.2 Problemas en el uso de passwords*, **pudiendo** no solo **comprometer la integridad de todo el sistema**, sino también los **datos de los usuarios** que pueda contener, ya que comúnmente las personas emplean un máximo de cinco passwords, a menudo derivadas entre sí, por lo que se pondría en peligro no solo la cuenta del sistema comprometido, sino también la de todos los sistemas en los que se autentique mediante la misma password.

Actualmente, si se habla de sistemas o infraestructuras sensibles, como puede ser un centro de supercomputación, o un hospital donde se almacenan miles de datos de pacientes, se tiende a pensar que el acceso a los mismos está muy protegido siendo muy complicado penetrar en dichos sistemas. Sin embargo, nada más lejos de la realidad, se suele emplear el mismo sistema de **usuario – password**, por lo que se analizará en los apartados siguientes el porqué de esta solución, o si se puede alcanzar o emplear alguna otra solución más eficiente y/o segura.

2.2. Autenticación mediante passwords y problemas en su uso

Una password consiste en un **conjunto de caracteres alfanuméricos** que proporciona acceso a un usuario en un servicio o infraestructura. Las passwords autentican el **par usuario – password** en dicho servicio o infraestructura, gracias a estar cierta información referente al **par** (en la mayoría de las ocasiones se almacena el usuario – password al completo) almacenado en el sistema al que se pretende acceder.

Los problemas que atañen al uso de las passwords [8] como sistema de autenticación pueden ser englobados en dos grandes apartados descritos a continuación; ***problemas por parte de los usuarios***, donde se refieren a fallos cometidos por éstos a la hora de introducir, crear y/o almacenar las passwords; y los ***problemas por parte de los sistemas de seguridad y las empresas***, donde se definirán los problemas que surgen a la hora de ser tratados y/o almacenados los datos de los usuarios en los distintos servicios que se pretenden proteger.

Por último se detallará en el apartado ***2.2.3. Gastos empresariales relacionados con el uso de passwords***, algunos ejemplos y aproximaciones de las inversiones que deben realizar las empresas; para mantener este sistema de seguridad aparentemente barato y sencillo de usar por parte de los usuarios, así como las pérdidas que se generan cada vez que un sistema se ve comprometido por el uso de las mismas.

2.2.1. Problemas por parte de los usuarios

El **primer problema** se plantea tras haber hablado del auge actual de las redes sociales y múltiples servicios web, los cuales aseguran sus sistemas por medio de passwords, derivando en que los **usuarios posean decenas de cuentas** que deben proteger. Esto implica que **la mayoría de las passwords se derivan entre sí**, o son cadenas de caracteres sencillas de recordar para ellos.

Esto no sería un problema de no ser por el crecimiento casi exponencial de la potencia de cálculo de las máquinas, tal y como predijo el cofundador de Intel, Gordon Moore.

"The number of transistors incorporated in a chip will approximately double every 24 months."

--Gordon Moore, Intel co-founder

Gracias a las máquinas actuales, cuya potencia de cálculo puede llegar hasta 33.86 Petaflops (esto son 33.86 * 10^{15} operaciones en coma flotante) tal y como se detalla en la lista de Noviembre de 2013 del top500 [9], se podría averiguar **en solo cuestión de segundos cualquier password** incorrectamente formada por medio de **ataques de fuerza bruta** sea cual sea el algoritmo criptográfico empleado para protegerla. Por este motivo, se recomienda que las passwords tengan un tamaño mínimo y una variedad de caracteres en su construcción.

El **segundo problema** que se puede identificar, proviene de las restricciones que se deben tener en cuenta a la hora de escoger un password, ya que es un hecho que una persona puede memorizar entre cuatro y nueve passwords correctamente formadas, esto es que posean entre 8 y 16 caracteres alfanuméricos donde se deben incluir letras tanto mayúsculas como minúsculas. Teniendo en cuenta la media de sistemas y servicios (identidades digitales) a proteger por medio de las mismas, se puede comprobar que hay un claro problema. Por este motivo, los usuarios tienden a la repetición de passwords, o a la malformación de las mismas para simplificarse el memorizarlas, lo que nos lleva de nuevo al **primer problema**.

El **tercer problema** que se ha considerado deriva del segundo, ya que viene a ser el hecho de que los usuarios repitan o debiliten sus passwords; pues implica que trasladen el problema a sistemas que puedan ser más sensibles, tanto para ellos como para la empresa o lugar de trabajo, pudiendo dar lugar a pérdidas económicas o de datos sensibles.

Pero a pesar de que se emplease una password suficientemente segura, y de que tan solo se emplease para proteger un sistema que el usuario considere sensible, como puede ser el acceso al lugar de trabajo, siempre existe el **cuarto problema**, donde se demuestra que **los usuarios (las personas) son el eslabón más débil en cualquier sistema de seguridad**. Este problema hace referencia a los innumerables medios que poseen los crackers para obtener dichas claves de acceso, como pueden ser **ataques de phishing**, la infección de alguna aplicación del usuario por medio de **troyanos**, o la inserción de **keyloggers** para capturar las pulsaciones realizadas en el teclado por el usuario víctima. Además, se debe tener en cuenta la posibilidad de que la password sea capturada visualmente por una tercera persona durante el proceso de escritura de la misma por parte del usuario.

2.2.2. Problemas por parte de los sistemas de seguridad y las empresas

En muchas ocasiones no es culpa de los usuarios, sino de la administración y gestión de los sistemas de autenticación llevada a cabo por las empresas que soportan los distintos servicios.

A pesar de que existen estándares para las buenas prácticas a la hora de tratar con los datos de los usuarios [10], como son las normas ISO 27001 e ISO 27002, no quiere decir que siempre sean aplicados. Se han dado muchos casos, como los citados en la bibliografía y publicados por el abogado Javier Hernández Martínez [11], donde las empresas, bien por error suyo o bien por poder disponer de información privilegiada, no almacenan o tratan los datos del usuario debidamente, esto es mediante el cifrado de los mismos. Esto puede desembocar en grandes fugas de datos cuando dichas empresas se ven afectadas por algún ataque informático, permitiendo que terceras personas accedan a dicha información; lo que acarrea en denuncias que oscilan entre 60.000 y 300.000 euros según el Real Decreto 1720/2007 en su artículo 93.4, vigente en el momento en que se dieron los casos analizados por el abogado, y que redacta:

"El documento de seguridad establecerá la periodicidad, que en ningún caso será superior a un año, con la que tienen que ser cambiadas las contraseñas que, mientras estén vigentes, se almacenarán de forma ininteligible".

Además se debe tener en cuenta que **para el debido almacenaje de las passwords** no basta con que se encuentren en un estado "ininteligible", es decir, a la hora de emplear **algoritmos criptográficos** se debe optar siempre por aquellos que han sido **públicamente probados,** y que además se consideran actualmente seguros. Actualmente, siguiendo las indicaciones del *National Institute of Standards and Technology* (NIST) en la publicación del 2011 [24], se podría considerar como **algoritmo criptográfico simétrico** seguro el **AES 256,** o el **RSA 2048** (se desaconseja RSA 1024 desde el año 2013) en caso de considerar un **algoritmo asimétrico**. Se debe descartar tanto algoritmos privativos, cuya metodología de cifrado no es mostrada al público y por tanto su seguridad no ha podido ser concienzudamente probada, como algoritmos ya considerados "rotos" como puede ser el **cifrado DES**. En caso de proteger datos sensibles con algoritmos de estos estilos el contenido **podría ser descifrado en cuestión de pocos minutos**.

Otro caso en que se podrían considerar errores de seguridad por parte de las empresas sería a la hora de tratar la autenticación del usuario de forma remota, es decir, en los métodos de introducción de las passwords, cuando el usuario pretende acceder a un sistema de forma remota. En este campo se podrían detallar ataques tipo **SQL Injection**, **DDOS** (Distributed Denial of Service) o incluso de nuevo los **ataques por fuerza bruta** si no se limitan los intentos de acceso.

Para terminar con este apartado se señalarán dos ejemplos donde se pudo comprobar tanto la magnitud del daño de un ataque informático, como la mala gestión de la información que se hace a día de hoy incluso en las grandes empresas, como son las que se detallan a continuación:

- *Caso PlayStation Network (PSN):*

  En Abril de 2011, todos los medios de comunicación centraron sus noticias en el **ataque DDoS** que sufrió la empresa **Sony Computer Entertaiment** [12], donde se vio afectado su servicio online para venta de contenidos digitales y soporte para sus principales plataformas de videojuegos (PSN). El ataque consistió en un ataque DDoS que, no solo dejó inoperativo el sistema durante días, sino que supuso el acceso temporal de terceras personas a los datos de aproximadamente **330.000 usuarios españoles y 70 millones** de usuarios **a nivel mundial**, por lo que se les aconsejó bloquear las tarjetas de crédito almacenadas en dicho servicio.

  La FACUA demandó a la Agencia Española de Protección de Datos (AEPD) para que determinase si Sony había vulnerado el **principio de seguridad de los datos,** que es regulado por el artículo 9 de la Ley Orgánica 15/1999, y donde se establece que:

  "El responsable del fichero** (que contiene los datos de los usuarios), y, en su caso, el encargado del tratamiento, **deberán adoptar las medidas de

índole técnica y organizativas necesarias que garanticen la seguridad de los datos de carácter personal y eviten su alteración, pérdida, tratamiento o acceso no autorizado**, habida cuenta del estado de la tecnología, la naturaleza de los datos almacenados y los riesgos a que están expuestos**”

De este suceso se debe destacar, no solo la **brecha de seguridad** que sufrió la compañía, sino la **mala gestión** que realizó la empresa tras el ataque, ya que encubrió los hechos durante tres días, sin informar a ningún usuario para que pudieran tomar precauciones. Además se tardó semanas en conseguir recuperar un servicio estable de cara a sus usuarios.

Los **gastos** sufridos por la empresa **no** han sido **calculados**, siendo miles los usuarios los que decidieron abandonar el servicio, a pesar de las muchas compensaciones que se ofreció por lo sucedido.

- *Caso Heartland Payment System (HPS):*

Esta empresa estadounidense, encargada de proveer créditos, administrar tarjetas de crédito y débito, así como pagos y transacciones tanto físicas como online, sufrió un ataque **SQL Injection** en el 2009 a manos de un cracker de 28 años [13], cuya intención era la de vender la información de los clientes que HPS administraba.

Heartland Payment System, que en aquel momento procesaba transacciones para más de 250.000 empresas, debió gastar 12,6 millones de dólares americanos para reparar y compensar a sus clientes por los problemas causados.

Según el periódico de ElMundo.es, tal y como se detalla en el enlace de la bibliografía referente al caso, las autoridades estadounidenses emitieron el siguiente comunicado:

"Se cree que el esquema constituye el mayor caso de hackeo y robo de identidades juzgado por el departamento de Justicia de EEUU"

El cracker en cuestión se enfrentó a 20 años de prisión, siendo juzgado además por otros delitos informáticos.

2.2.3. Gastos empresariales relacionados con ataques informáticos

Si bien es cierto que el apartado 2.2 trata acerca de las passwords y los problemas relacionados con ellas, este sub-apartado se ha querido redactar con la intención de concienciar acerca de los gastos que le supone a las empresas el sufrir un ataque informático, que aunque no tiene por qué derivar de un robo de password, siempre se relaciona con el acceso a la información de la empresa; por lo que directa o indirectamente los gastos que se comentarán en este sub-apartado se relacionan

con los problemas que acarrean los fallos en los sistemas de acceso a dicha información.

Se han evaluado diversos **informes estadísticos** [14] distribuidos por algunas de las grandes empresas globales en seguridad, como son **McAfee** y **Kaspersky Lab**, de donde se han obtenido tablas y gráficos con contenido referente al año en curso (2013).

Se conoce que los **mayores gastos** de cara a las empresas se producen cuando los **ataques no son identificados y tratados a tiempo**, es decir, cuando al no detectar que se está sufriendo un ataque, los atacantes pueden ir recopilando información e ir ascendiendo privilegios y ganando niveles de acceso dentro de la empresa víctima.

A mediados del presente año, McAfee publicó en su informe "***Needle in a data risk: the rise of big security data***" cómo tan solo el 35% de 500 compañías globales entrevistadas poseían la capacidad de identificar y controlar una filtración en su sistema en los primeros minutos en que se produce. Además informó de que el 22% requiere de al menos un día para detectarlo, y el 5% al menos una semana para detectar amenazas, tiempo más que suficiente para que las pérdidas asciendan al equivalente de miles de euros.

McAfee concluye su informe diciendo que *"además de la capacidad de divisar las amenazas en tiempo real, las organizaciones deben tener la capacidad de identificar las tendencias y los patrones potencialmente siniestros en el largo plazo. Más allá de encontrar una 'aguja en una pila de datos', las organizaciones deben ampliar el plazo con un contexto basado en riesgos, para encontrar la aguja correcta y poder lidiar, así, en forma proactiva con las amenazas actuales".*

Por otro lado, el **Center for Strategic and International Studies (CSIS)** publicó en su informe "***The Economic Impact of Cybercrime and Cyber Espionage***" un enfoque cuantitativo basado en los datos del Departamento de Comercio y las pérdidas análogas, como son los accidentes automovilísticos, piratería y demás crímenes tal y como se muestra en la *imagen 1.*

Putting Malicious Cyber Activity in Context

CRIMINAL ACTION	ESTIMATED COST	PERCENT OF GDP	SOURCE
GLOBAL			
Piracy	$1 billion to $16 billion	0.008% to 0.02%	IMB
Drug Trafficking	$600 billion	5%	UNODC
Global cyber activity	$300 billion to $1 trillion	0.4% to 1.4%	Various
US ONLY			
Car Crashes	$99 billion to $168 billion	0.7% to 1.2%	CDC, AAA
Pilferage	$70 billion to $280 billion	0.5% to 2%	NRF
US- cyber activity	$24 billion to $120 billion	0.2% to 0.8%	Various

Imagen 1, tabla extraída del informe del CSIS sobre gastos por ataques informáticos

El informe del CSIS ha sido de los primeros en estimar los gastos económicos causados por delitos informáticos incluyendo la delincuencia, la pérdida de la propiedad intelectual, la pérdida de reputación, los costos de reforzar la seguridad, y los propios gastos de la recuperación tras el ataque.

El CSIS analizó a través de los datos del Departamento de Comercio la pérdida de puestos de empleo, estimando en 508.000 puestos de trabajo las pérdidas generadas por el espionaje cibernético.

La empresa Kaspersky Lab ha estimado también en este año 2013, y según los datos recogidos en el documento "***Encuesta Global sobre seguridad TI corporativa - 2013***", llevada a cabo por ellos mismos junto a B2B Internacional, que el **coste medio en el que incurren las empresas** como consecuencia de un ataque informático ronda los **500.000 euros,** en función de la región geográfica en que opere la empresa en cuestión. Sin embargo, y como se puede observar en la *imagen 2* recogida de dicho informe, los costes a los que se somete la pequeña y mediana empresa de cara a afrontar un ataque informático es de aproximadamente 38.000 euros, yendo en correlación con las ganancias que obtienen.

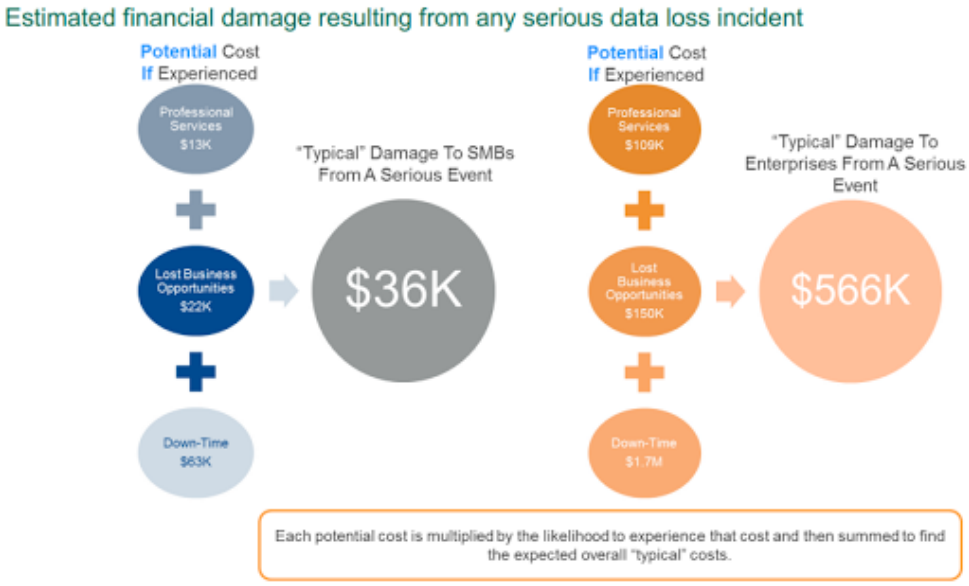

Imagen 2, gráfico obtenido del informe realizado por Kaspersky Lab

Con todos estos detalles extraídos y analizados de los informes ya mencionados, se puede concluir que los delitos informáticos implican un altísimo coste de cara a una empresa o al propio país. Además de que todos estos ataques, se ven facilitados por los errores o problemas que trae el emplear sistemas de autenticación que puedan dar lugar a vulnerabilidades en la seguridad de la empresa; como son los clásicos métodos de introducción de passwords analizados en el apartado que se está concluyendo.

En los siguientes apartados se analizarán, y evaluarán, los sistemas y métodos de autenticación que no dependen de la introducción de un password por parte del

usuario. De esta forma se podrían evitar los numerosos problemas que se ha visto que pueden generar tanto involuntariamente; siendo el blanco de un ataque o pérdida de la password; como voluntariamente, ya que muchos ataques producidos en empresas se deben a miembros internos de la empresa, a los que no se les ha restringido todos sus accesos tras un despido o situación similar, permitiéndoles acceder a información privilegiada que pueden vender en tono de venganza.

2.3. Autenticación mediante firmas y certificados (criptografía asimétrica)

En los siguientes apartados se tratará de definir de la forma más concisa posible los diferentes aspectos que conforman la criptografía asimétrica.

La criptografía asimétrica descarta el principal problema que posee la criptografía simétrica, que no es otro que el intercambio de claves entre los usuarios que desean comunicarse entre sí.

Sin entrar en detalles explicativos acerca de este tipo de criptografía en este apartado, la criptografía asimétrica es actualmente uno de los mecanismos de seguridad más fiables, pues consigue tanto la **identificación** como la **autenticación** del remitente de un mensaje.

Cada participante de una comunicación posee **dos claves**, una **pública** y otra **privada**, pudiéndose emplear de dos formas:

- En caso de que se quiera **cifrar el mensaje, ocultando así su contenido** de cara a terceras personas, no es necesario el establecimiento de una clave común entre el emisor y el receptor de la conversación; basta con que el emisor cifre el mensaje con la clave pública del receptor, de forma que sólo el receptor con su clave privada podrá descifrar el contenido del mensaje (siempre y cuando nadie se la haya robado).

- Por otro lado, en caso de que la idea sea **garantizar la autenticidad del emisor,** basta con que él mismo firme el mensaje con su clave privada, por lo que cualquiera que posea su clave pública podrá verificar que es quien dice ser; pues a excepción de que se le haya sustraído la clave privada, él debe ser el único poseedor de su clave privada.

En este trabajo es interesante la evaluación de este tipo de criptografía, ya que como se ha mencionado en el segundo punto acerca de los posibles usos de la criptografía asimétrica, se puede garantizar la autenticidad del emisor de cara a un usuario o un servidor (como es nuestro caso) mediante el uso de este tipo de claves. Este modo de uso de la criptografía asimétrica es también conocido como ***firma digital***, y será debidamente explicado en su correspondiente apartado *2.3.2. Firma digital*.

2.3.1. Estándar X.509

El estándar X.509 [15] es un estándar empleado en criptografía para especificar los formatos que deben poseer los certificados de claves públicas; principalmente **empleados en infraestructuras de claves públicas o PKI,** tal y como se detallará en el apartado *2.3.3. Infraestructura de clave pública*.

Su sintaxis es definida mediante el lenguaje ASN.1 (Abstract Syntax Notation One [16]), y los formatos de codificación que comúnmente son empleados son CER, DER (Distinguish Encoding Rules) o PEM (Privacy Enhanced Mail).

Fue publicado oficialmente en el 1988, y se apoya en un sistema jerárquico donde los certificados son emitidos estrictamente por autoridades certificantes o **CA**s, las cuales pueden firmar claves públicas y por tanto dar validez a los certificados de claves de otras personas o entidades.

Una CA es una entidad capaz de emitir certificados digitales para el uso de terceros, es decir, es una entidad que **se supone confiable en un entorno** como puede ser el de una empresa u organización. Un ejemplo podrían ser los certificados raíz de una organización, los cuales son distribuidos a los miembros de dicha organización para acceder a una infraestructura PKI. Otro ejemplo es el de las CAs incluidas en navegadores web por defecto, de manera que los certificados SSL de grandes empresas que han pagado por este privilegio funcionen al instante.

Aun así un tercero puede decidir que CA es de confianza, pudiendo acceder a conexiones seguras proporcionadas por organizaciones concretas. Por otro lado este hecho da lugar a los ya mencionados, y explicados en el *apéndice II*, ataques MiTM.

Como se puede observar en los dos anexos incluidos al final de este sub-apartado, las **partes de que se compone un certificado X.509**, a grandes rasgos, son:

- ***Versión*** del certificado.
- ***Número de serie***.
- ***Identificador del algoritmo*** de cifrado empleado.
- ***Datos del emisor***, donde entre otros se encuentran los referentes al nombre de la organización (O), el common name (CN), la unidad organizativa (OU) y el país (C).
- ***La fecha de validez*** del certificado en cuestión.

- ***Datos del sujeto***, donde en caso de ser un certificado auto firmado coincidirá con los datos del emisor.

- ***Información de la clave pública*** del sujeto, donde se proporciona información del algoritmo empleado en la clave pública, así como la clave pública del mismo.
- ***Firma de la CA***, donde en el primer anexo se puede comprobar que ha sido formada tomando un hash MD5 de la primera parte del certificado, cifrándolo posteriormente con la clave privada RSA (*md5withRSAEncryption*) de la CA, en este caso Thawte, por lo que poseyendo la clave pública de dicha CA se puede verificar que el certificado es válido.

```
Certificate:
   Data:
       Version: 1 (0x0)
       Serial Number: 7829 (0x1e95)
       Signature Algorithm: md5WithRSAEncryption
       Issuer: C=ZA, ST=Western Cape, L=Cape Town, O=Thawte Consulting cc,
               OU=Certification Services Division,
               CN=Thawte Server CA/Email=server-certs@thawte.com
       Validity
           Not Before: Jul  9 16:04:02 1998 GMT
           Not After : Jul  9 16:04:02 1999 GMT
       Subject: C=US, ST=Maryland, L=Pasadena, O=Brent Baccala,
                OU=FreeSoft, CN=www.freesoft.org/Email=baccala@freesoft.org
       Subject Public Key Info:
           Public Key Algorithm: rsaEncryption
           RSA Public Key: (1024 bit)
               Modulus (1024 bit):
                   00:b4:31:98:0a:c4:bc:62:c1:88:aa:dc:b0:c8:bb:
                   33:35:19:d5:0c:64:b9:3d:41:b2:96:fc:f3:31:e1:
                   66:36:d0:8e:56:12:44:ba:75:eb:e8:1c:9c:5b:66:
                   70:33:52:14:c9:ec:4f:91:51:70:39:de:53:85:17:
                   16:94:6e:ee:f4:d5:6f:d5:ca:b3:47:5e:1b:0c:7b:
                   c5:cc:2b:6b:c1:90:c3:16:31:0d:bf:7a:c7:47:77:
                   8f:a0:21:c7:4c:d0:16:65:00:c1:0f:d7:b8:80:e3:
                   d2:75:6b:c1:ea:9e:5c:5c:ea:7d:c1:a1:10:bc:b8:
                   e8:35:1c:9e:27:52:7e:41:8f
               Exponent: 65537 (0x10001)
   Signature Algorithm: md5WithRSAEncryption
       93:5f:8f:5f:c5:af:bf:0a:ab:a5:6d:fb:24:5f:b6:59:5d:9d:
       92:2e:4a:1b:8b:ac:7d:99:17:5d:cd:19:f6:ad:ef:63:2f:92:
       ab:2f:4b:cf:0a:13:90:ee:2c:0e:43:03:be:f6:ea:8e:9c:67:
       d0:a2:40:03:f7:ef:6a:15:09:79:a9:46:ed:b7:16:1b:41:72:
       0d:19:aa:ad:dd:9a:df:ab:97:50:65:f5:5e:85:a6:ef:19:d1:
       5a:de:9d:ea:63:cd:cb:cc:6d:5d:01:85:b5:6d:c8:f3:d9:f7:
       8f:0e:fc:ba:1f:34:e9:96:6e:6c:cf:f2:ef:9b:bf:de:b5:22:
       68:9f
```

Anexo 1. Ejemplo certificado X509 de *www.freesoft.org* tomado de Wikipedia

```
Certificate:
   Data:
       Version: 3 (0x2)
       Serial Number: 1 (0x1)
       Signature Algorithm: md5WithRSAEncryption
       Issuer: C=ZA, ST=Western Cape, L=Cape Town, O=Thawte Consulting cc,
               OU=Certification Services Division,
               CN=Thawte Server CA/Email=server-certs@thawte.com
       Validity
           Not Before: Aug  1 00:00:00 1996 GMT
           Not After : Dec 31 23:59:59 2020 GMT
       Subject: C=ZA, ST=Western Cape, L=Cape Town, O=Thawte Consulting cc,
                OU=Certification Services Division,
                CN=Thawte Server CA/Email=server-certs@thawte.com
       Subject Public Key Info:
           Public Key Algorithm: rsaEncryption
           RSA Public Key: (1024 bit)
               Modulus (1024 bit):
                   00:d3:a4:50:6e:c8:ff:56:6b:e6:cf:5d:b6:ea:0c:
                   68:75:47:a2:aa:c2:da:84:25:fc:a8:f4:47:51:da:
                   85:b5:20:74:94:86:1e:0f:75:c9:e9:08:61:f5:06:
                   6d:30:6e:15:19:02:e9:52:c0:62:db:4d:99:9e:e2:
                   6ª:0c:44:38:cd:fe:be:e3:64:09:70:c5:fe:b1:6b:
                   29:b6:2f:49:c8:3b:d4:27:04:25:10:97:2f:e7:90:
                   6d:c0:28:42:99:d7:4c:43:de:c3:f5:21:6d:54:9f:
                   5d:c3:58:e1:c0:e4:d9:5b:b0:b8:dc:b4:7b:df:36:
                   3a:c2:b5:66:22:12:d6:87:0d
               Exponent: 65537 (0x10001)
       X509v3 extensions:
           X509v3 Basic Constraints: critical
               CA:TRUE
   Signature Algorithm: md5WithRSAEncryption
       07:fa:4c:69:5c:fb:95:cc:46:ee:85:83:4d:21:30:8e:ca:d9:
       a8:6f:49:1ª:e6:da:51:e3:60:70:6c:84:61:11:a1:1ª:c8:48:
       3e:59:43:7d:4f:95:3d:a1:8b:b7:0b:62:98:7ª:75:8ª:dd:88:
       4e:4e:9e:40:db:a8:cc:32:74:b9:6f:0d:c6:e3:b3:44:0b:d9:
       8ª:6f:9ª:29:9b:99:18:28:3b:d1:e3:40:28:9ª:5ª:3c:d5:b5:
       e7:20:1b:8b:ca:a4:ab:8d:e9:51:d9:e2:4c:2c:59:a9:da:b9:
       b2:75:1b:f6:42:f2:ef:c7:f2:18:f9:89:bc:a3:ff:8a:23:2e:
       70:47
```

Anexo 2. Ejemplo de certificado auto firmado de *Thawte* tomado de Wikipedia

En este último anexo se puede observar el claro ejemplo de un certificado auto firmado, ya que como se ha dicho anteriormente, coindice el CN del emisor con el del sujeto. En este caso la única forma de validar la firma es contrastándola contra sí misma, por lo que la única forma de hacer este certificado confiable es configurarlo manualmente, es decir, el usuario debe confiar en dicha organización.

2.3.2. Firma digital

El sistema de firma digital es empleado, tal y como se ha ido mencionando, para verificar la identidad del emisor de un mensaje en una comunicación.

En la *imagen 3* se puede observar de forma gráfica un ejemplo de comprobación del sistema de firma digital, el cual se apoya en el esquema del estándar de certificación ya mencionado en el apartado *2.3.1. Estándar X.509*.

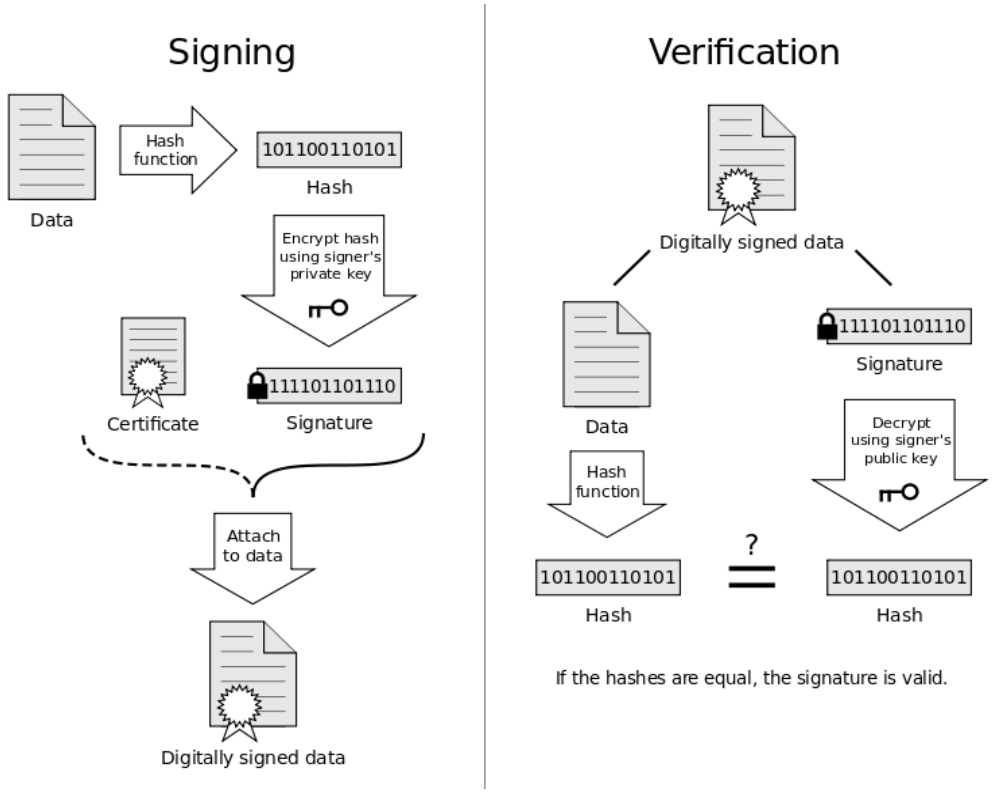

Imagen 3, explicación de firma digital obtenida de Wikipedia

El proceso de firma consiste en los siguientes pasos:

1) El emisor **redacta un mensaje en claro,** al cual se le **aplica** una función resumen o ***función hash***. Algún ejemplo de función hash empleada actualmente puede ser el algoritmo SHA-1 (160 bits) o aún más efectivo el SHA-256 (256 bits).

2) Se **cifra el hash** obtenido **mediante** su **clave privada**. El algoritmo empleado debe ser un algoritmo de cifrado asimétrico, como por ejemplo el cifrado RSA o ElGamal, si se busca el cifrado del contenido del mensaje; o DSA si tan solo se busca la firma, y por tanto autenticidad, del mensaje.

3) El **emisor envía** al receptor el **cuerpo del mensaje** en claro, o cifrado, en cuyo caso se cifra con la clave pública del receptor. Esto se adjunta con la **firma del mensaje**, compuesta por el *hash* obtenido en el *paso 1* y el certificado digital del emisor, donde se incluyen los datos personales del

mismo junto a las claves empleadas, tal y como se detalla en el apartado *2.3.1 Estandar X.509.*

La recepción, y por tanto verificación del mensaje por parte del receptor, se compone de los siguientes pasos o etapas:

1) Con los datos recibidos por parte del emisor, el receptor debe verificar la autenticidad del certificado digital del emisor en caso de que posea una CA superior, tal y como se detalló en el apartado anterior.

2) Una vez se confíe en el certificado del emisor, el receptor puede acceder a la clave pública del emisor, empleándola para descifrar el *hash* del mensaje.

3) Aplicando la función *hash,* que se detalla en el certificado del emisor al cuerpo del mensaje ya en claro, el receptor compara el resultado con el obtenido en el punto anterior mediante el descifrado con la clave pública del emisor.

4) En caso de que los "*hashes"* coincidan quiere decir que el mensaje no ha sido alterado, es decir, que el emisor es quien dice ser, obteniéndose así la información sobre quién es el emisor.

Como recapitulación de este sub apartado se tiene que, empleando criptografía asimétrica, se obtiene la finalidad que se busca con el uso de la firma digital, la cual se puede dividir en tres apartados:

- ***Autenticación:*** Mediante la firma digital se puede asegurar que el emisor es quien dice ser, pues como se ha explicado en este sub apartado, y se puede observar gráficamente en la *imagen 3*, el mensaje a enviar es cifrado mediante la clave privada del emisor, para ser descifrada con su correspondiente clave pública por parte del receptor. De esta forma se consigue verificar que solo puede ser el auténtico emisor quien ha enviado ese mensaje, pues la clave privada solo puede estar en posesión del mismo emisor.

- ***Integridad:*** Dado que el contenido del mensaje es cifrado con la clave privada del emisor, y al receptor se le envía tanto la firma como el cuerpo del mensaje, en caso de que se modifique su contenido por parte de una tercera persona ajena a la comunicación, el receptor natural de la conversación podrá darse cuenta de este hecho. Cuando el receptor intente verificar el *hash* del contenido con la clave pública del emisor, éste no coincidirá con el *hash* resultante al aplicar la función resumen sobre el

cuerpo del mensaje, es decir, cualquier cambio en el mensaje tras ser firmado invalidará la propia firma.

- ***No repudio:*** Con todo ya detallado es fácil discernir que, dado que tras firmar un mensaje, el emisor solo puede ser una persona concreta, ésta nunca podrá negar el haber enviado dicho mensaje.

Este sistema de verificación y autenticidad de los participantes en una comunicación, no solo es válido para el envío de mensajes, como se ha podido dar a entender en los ejemplos comentados; sino que es la base del establecimiento de cualquier canal seguro empleado actualmente, como vienen siendo los protocolos SSL y TLS, así como también sirve de base para el acceso a un sistema o infraestructura de la misma forma que se hace con el uso de passwords.

El sistema de firma digital es, además, la base para los actuales sistemas de autenticación basados en directorios explicados en siguientes apartados, como son las infraestructuras de claves públicas o PKI, directorios LDAP, y Kerberos.

2.3.3. Infraestructura de clave pública (PKI)

Una vez se tiene conocimiento de los conceptos básicos de que se compone una **infraestructura de clave pública** (PKI), se puede proceder a la explicación de la propia infraestructura.

La infraestructura de clave pública es la combinación de varios elementos que se verán a continuación, donde empleando el sistema de firma digital, es posible garantizar los tres aspectos mencionados en el apartado anterior *2.3.2. Firma digital*, que no son otros que la autenticidad, integridad y el no repudio de un usuario en una comunicación.

Esta infraestructura se emplea por tanto no solo en el aspecto que nos atañe, la autenticación de usuarios y sistemas, sino también para cifrar mensajes, firmarlos y, en combinación con algoritmos de criptografía simétrica o asimétrica, crear comunicaciones seguras como son los protocolos SSL (Secure Socket Layer) y TLS (Transport Layer Security). Estos últimos, detallados en el apartado *9.1. Comunicaciones seguras* del *apéndice I*, son empleados para asegurar las comunicaciones entre cliente y servidor, lo que en nuestro caso ya no solo nos proporciona una autenticación entre las distintas partes de una comunicación, sino que dificulta que terceras personas puedan interceptar la comunicación sin necesidad de obtener el certificado de acceso del usuario, es decir, es otro punto favorable, pues asegura los accesos remotos.

En la *imagen 4* se puede apreciar un gráfico en el que se muestran las tres partes principales que intervienen y conforman una infraestructura de clave pública:

- **CA (Certificate Authority):** Las autoridades de certificación, tal y como se detalló en el apartado *2.3.1 Estándar X.509*, son las encargadas de firmar las claves públicas generadas por los usuarios. Genera certificados de

forma que otras personas puedan confiar y verificar la identidad del usuario. Como ya se mencionó, la CA debe ser de confianza, aunque también se pueden generar CAs propias, donde los clientes deberán confiar en ellas, bien por que conozcan la entidad personalmente, o porque a través de otras vías sepan de la veracidad de la misma.

- **RA (Registration Authority):** Son empleados sobretodo en infraestructuras grandes, donde la CA puede llegar a saturarse, para aliviar su función de generación de certificados.
  La RA se encarga de realizar un filtrado de los datos y peticiones de firmado realizadas por los usuarios, ordenándolas y enviándolas posteriormente a la CA para que firme las claves de los usuarios, es decir, en una infraestructura con RA el usuario debe enviarle la clave a firmar junto con el resto de trámites necesarios, recibiendo éste posteriormente su certificado con su clave firmada.

- **VA (Verification Authority):** Es la parte encargada de almacenar las claves públicas firmadas por la CA para, posteriormente, suministrarlas a quien las solicite. Además la VA es quien almacena las CRLs (Certificate Revocation List) donde se listan los certificados que han expirado o han sido revocados, de forma que no haya lugar a dudas de cara a posibles fraudes, y evitando así posibles fallas de seguridad por parte de usuarios cuyo acceso al sistema ha sido revocado.

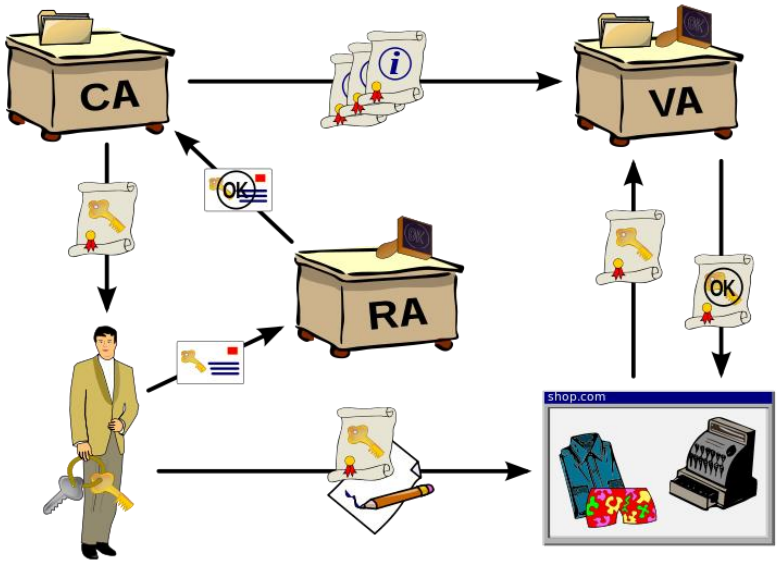

Imagen 4, explicación sobre PKI tomada de Wikipedia

La imagen anterior puede servir de apoyo al que sería un posible escenario de acceso a un servidor de compras, el cual posee una infraestructura de clave pública con las partes ya mencionadas:

1) El usuario, tras generar su par de clave pública – privada, envía su clave pública al RA, tramitando los formularios de identificación que el servicio le

pida para la generación del certificado.

2) Tras ordenar y filtrar la petición del usuario con sus datos, entre las correspondientes con otros usuarios, la RA le enviará la clave pública del usuario con sus datos a la CA.

3) La CA, habiendo recibido todo lo necesario, genera el certificado del usuario y envía la clave del mismo a la VA, para que lo almacene a la vez que envía al usuario su certificado correspondiente.

4) Una vez el usuario posee su certificado ya puede acceder al servicio de compras. Basta con que en su navegador (en caso de acceder de tal forma) tenga importado su certificado, con lo que cuando intente acceder al servicio, el servidor del sistema le pedirá el certificado que ha sido generado previamente.

5) El servidor enviará el certificado del usuario a la VA para que verifique que se encuentra en la lista y que no ha expirado o ha sido revocado. Si todo está en orden, el servidor concederá el acceso al usuario en el sistema.

En la mayoría de los casos, este proceso se emplea también para la generación de una comunicación segura entre cliente y servidor. Sin entrar en detalles, pues para ello se puede acceder al apartado *9.1. Comunicaciones seguras* del *apéndice I*, mediante este sistema, el cliente y servidor pueden intercambiar las claves de algoritmos de cifrado simétrico en lo que se llama la fase de saludo o *handshake* de los actualmente empleados protocolos SSL y TLS. Tras este proceso, el cliente y servidor ya pueden comunicarse cifrando toda su comunicación entre sí con cifrados simétricos, que son más rápidos y ligeros que los cifrados asimétricos.

2.4. Lightweight Directory Access Protocol (LDAP)

Tal y como el nombre indica, LDAP es un protocolo cliente – servidor para acceder a servicios de *directorio*, concretamente basados en certificados X.500. LDAP opera sobre TCP/IP, así como otros servicios de transferencia orientados a conexión. Su definición detallada y completa se encuentra en el **RFC3377**.

El concepto de **directorio** es similar al de una base de datos, pero suele contener una descripción más extensa e información basada en atributos, a menudo expresados con palabras nemotécnicas (distinguished name (DN) o common name (CN)). Al igual que una base de datos, un directorio debe ser capaz de responder lo más rápido posible a cualquier petición de búsqueda por parte de los clientes, independientemente del volumen de datos que posea almacenados. Además LDAP posee mecanismos de sincronización para evitar inconsistencias a causa de réplicas de los directorios internos.

El **almacenaje de la información** se realiza en estructuras con forma de árbol, donde la raíz de la jerarquía se encuentra en la cima. El nombramiento de los distintos nodos que componen el árbol se puede realizar de forma tradicional, o bien mediante

los dominios que pueda poseer la empresa en Internet, tal y como se puede ver en la *imagen 5* e *imagen 6*.

Conociendo la forma en que se almacena la información, y que LDAP opera de la misma manera que haría una base de datos, es fácil deducir que a la hora de realizar consultas, se pueden emplear los distintos nodos que componen el árbol a modo de filtros; además de los muchos tipos de consultas que se le puede realizar.

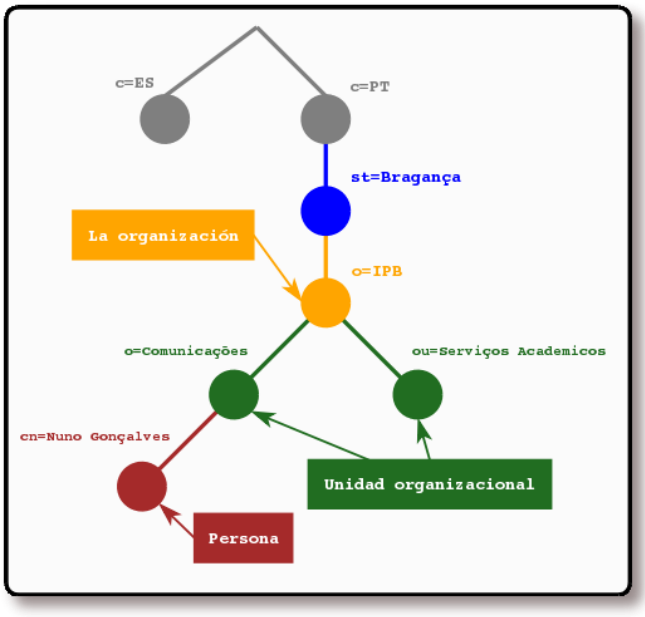

Imagen 5, ejemplo de jerarquía basada en el nombramiento tradicional

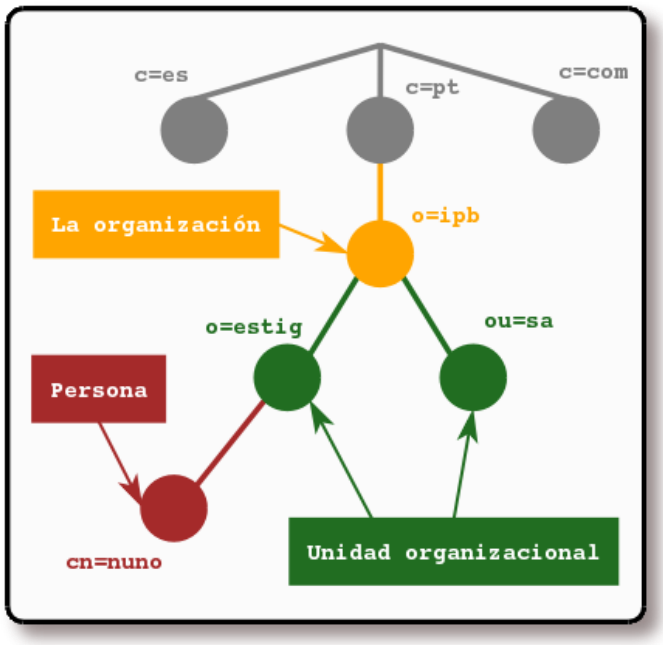

Imagen 6, ejemplo de jerarquía basada en el nombramiento de Internet

Como ya se ha mencionado, LDAP opera según el modelo cliente – servidor, por lo que en sus últimas versiones (LDAPv3) adopta mecanismos de paralelización y replicación de la información entre los distintos servidores LDAP de una misma organización.

2.4.1. Seguridad y autenticación en LDAP

Tras haberse explicado el concepto y funcionamiento de LDAP a modo de organizador de infraestructuras, conviene explicar de qué forma LDAP proporciona y controla el acceso a los usuarios a las distintas zonas o servicios, que el propio servidor o servidores LDAP administran.

Por defecto, el acceso a los distintos grupos de la organización (los nodos del árbol jerárquico) es abierto, es decir, que cualquier usuario de la empresa podría acceder a cualquier área. Para impedir esto se construyen las ACL (Access Control List), donde se definen las políticas de seguridad y acceso, restringiendo así el acceso según deseen los administradores. Para esto, se puede modificar de forma "sencilla" en los distintos servidores el archivo de configuración **slapd.conf**, restringiendo el acceso a nodos según grupos o miembros, y otorgándoles los permisos que el propio administrador desee. **Con las ACLs LDAP puede gestionar distintos niveles de acceso dentro del sistema o infraestructura**, emulando a los permisos de los distintos sistemas UNIX.

LDAP permite además gestionar el acceso al sistema por parte de los usuarios. Para ello se apoya en OpenSSH y OpenSSL (como ejemplos de desarrollo gratuito), de esta forma se pueden firmar y generar tanto certificados como pares de claves pública y privada, que posteriormente serán añadidos y gestionados por el servidor *slapd*. LDAP no posee internamente control de acceso, sino que se apoya en la premisa de la criptografía asimétrica de la autenticación mediante certificados, que previamente habrán de ser generados por medio de OpenSSH, por dar un ejemplo.

Además de todo esto, *slapd*, que no es más que un servidor de directorio LDAP, provee sistemas de protección e integridad mediante el uso del protocolo TLS. Para poder proporcionarlo necesita de certificados, como ya se ha mencionado.

Sin entrar en detalles de la configuración e instalación del servidor de directorios, pues hay multitud de libros [1] y enlaces para su correcta configuración (http://olsacupy.berlios.de/html/openldap-autentificacion-usuarios.html), se debe concluir este apartado haciendo mención a que la mayor parte de los administradores, así como papers y libros de consulta, prefieren y ven **más seguro el fortalecimiento de la autenticación en LDAP por medio de Kerberos**, explicado en el apartado a continuación. El motivo de esta decisión radica en la complejidad que posee LDAP para una correcta configuración, que se debe realizar para poseer un servicio de autenticación seguro; además Kerberos proporciona un sistema de autenticación más completo, proporcionando autenticación tanto de cliente como de servidor.

2.5. Protocolo Kerberos

Kerberos es un protocolo diseñado en el MIT (Masachussets Institute of Technology) como un protocolo de autenticación segura en la red. Fue diseñado para proveer de un sistema de autenticación fuerte a aplicaciones cliente – servidor empleando criptografía simétrica, e intentando solucionar los problemas que ésta posee (el intercambio de claves).

Kerberos se apoya en el **protocolo de clave simétrica de Needham-Schroeder**, empleando un tercero de confianza denominado como **centro de distribución de claves** (KDC, *Key Distribution Center*), que se compone a su vez de un **servidor de autenticación** (AS, *Authentication Server*) y un **servidor de tickets** (TGS, *Ticket Granting Server*). Kerberos mantiene una **base de datos** con las claves secretas, ya que cada entidad (cliente y servidor) comparte una clave secreta conocida únicamente por Kerberos y el mismo.

El protocolo Kerberos fue diseñado a fin de solucionar los problemas del **NIS** (Network Information Service), conservando además las virtudes que en su diseño alberga. Tal y como el MIT propaga [2], los defectos del NIS que a su vez solucionaron con Kerberos, es decir, la motivación con que se desarrolló el protocolo, son las siguientes:

- Evitar la propagación insegura de los datos de autenticación de los usuarios. A menudo un atacante, puede obtener los datos de autenticación de un usuario víctima mediante un sniffer o ataques MiTM, vulnerando así la seguridad de la red o de un sistema. Kerberos evita este problema, pues las passwords nunca cruzan la red, incluso en el primer login.

- En un entorno NIS no hay forma de identificar si un cliente es quien dice ser, es decir, son sistemas vulnerables a la suplantación de usuarios. Kerberos intenta solucionar este problema basándose en que la clave del usuario **solo** la posee el propio usuario (actúa a modo de clave privada).

- En otros entornos, cada vez que un usuario quiere volver a acceder al sistema debe introducir nuevamente su password, lo que aumenta las probabilidades de que sea espiado. En Kerberos evitan esto gracias al sistema de tickets que el protocolo posee.

Tal y como también destaca el MIT en su paper, estos problemas detallados no solo se encuentran ligados al NIS, sino también a otros servicios como puede ser el LDAP, véase como ejemplo el problema de la autenticación mediante texto en claro en la configuración por defecto de LDAP. Además, se destaca que Kerberos no solo crea canales de comunicación seguros, como SSL, sino que además autentica a los usuarios entre sí en el sistema.

2.5.1. Funcionamiento de Kerberos

Las comunicaciones en este protocolo se basan en **tickets**, siendo éstos almacenados por los clientes y distribuidos de forma cifrada por la red.

La parte principal de Kerberos es el KDC, el cual está constituido por tres partes:

- El **servidor de autenticación** (AS, *Autentication Server*), siendo el encargado de atender las peticiones de autenticación de los clientes. Emite los llamados *ticket granting ticket* (TGT).

- El **ticket granting server**, que es el encargado de emitir los conocidos como *ticket granting service* (TGS) a los clientes, lo que les proporciona autenticación de cara a un servicio o sistema de una red.

- La **base de datos**, donde se almacenan tanto las claves secretas de los clientes, como la de los servicios que mantiene Kerberos. Además, almacena información relativa a las cuentas de Kerberos (fecha de creación, políticas…). La base de datos debe ser mantenida y almacenada en un servidor independiente, evitando así que pueda comprometerse todo el sistema de forma más sencilla.

A continuación, se definirá la forma en que Kerberos trata la autenticación de los clientes de cara a un servicio. Se ha de recordar que **Kerberos tan solo proporciona autenticación, y no autorización** (no dispone de control de permisos en las distintas zonas del sistema a autenticar, ni información de *home* de usuarios). Si se desea disponer de servicios de autorización y organización, se debe implementar junto a Kerberos un sistema como puede ser NIS o LDAP, es decir, Kerberos se complementa con este tipo de sistemas a la perfección.

El protocolo Kerberos se puede dividir en dos grandes procesos, cada uno dependiente de uno de los servidores de que se compone el KDC. Se denominará a estos dos procesos como **mecanismo de autenticación** (dependiente de los *ticket granting tickets*) y **mecanismo de uso del servicio** (dependiente del *ticket granting service*).

Como **primera instancia** el usuario deberá obtener un *ticket granting ticket*, que le proporcionará el posterior acceso al servicio en que se quiere autenticar, sin necesidad de introducir su contraseña en múltiples ocasiones. Un TGT proporciona al cliente la forma de pedir acceso para los determinados servicios que sean protegidos por el protocolo, siempre y cuando tenga validez y no haya caducado (su duración suele ser de cinco a diez minutos).

Para obtener este tipo de ticket, el usuario deberá enviar una petición *request* al KDC, concretamente al AS. El servidor de autenticación, tras comprobar en la base de datos que el usuario existe, envía el TGT cifrado con la clave del usuario que estaba almacenada en la base de datos de Kerberos. Por último el usuario, quien posee la clave para descifrar el mensaje, obtiene el TGT y una clave de sesión de corta duración para la posterior petición del TGS.

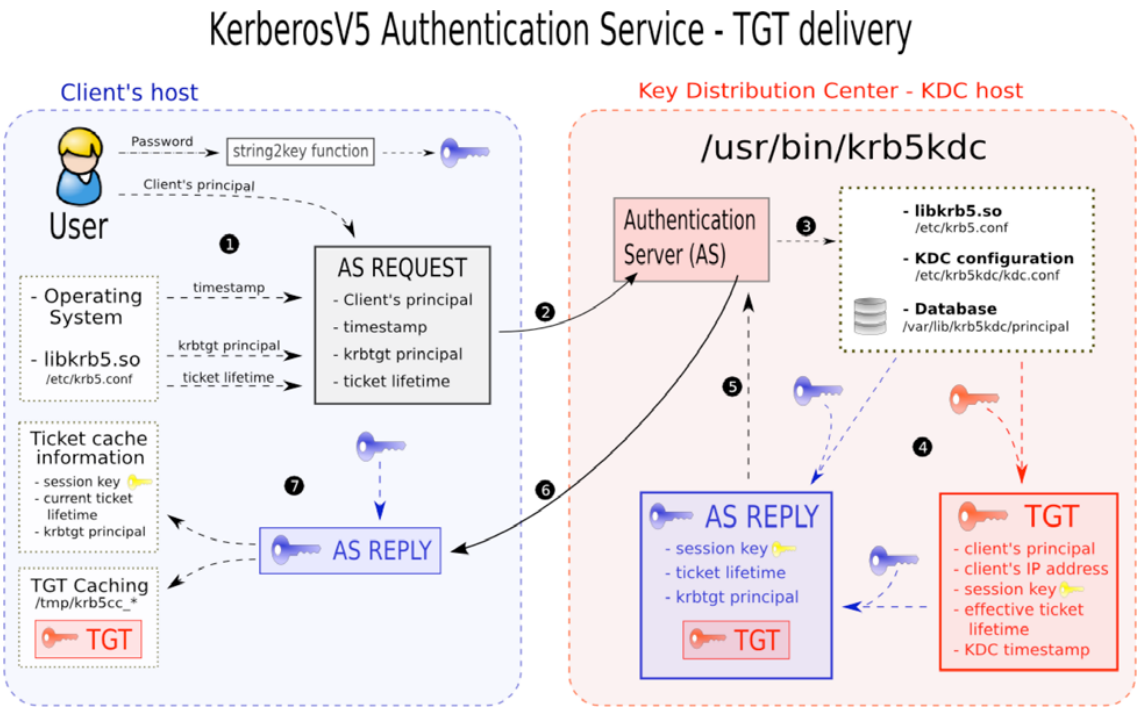

Imagen 7, descripción gráfica del proceso de obtención del TGT, obtenida de Wikipedia

Como **segunda instancia**, donde el usuario ya posee un **TGT válido**, el usuario deberá obtener un TGS para acceder a un servicio "kerberizado" (esto es, controlado su acceso por el protocolo Kerberos). El usuario enviará una petición al *Ticket Granting Server*, la cual se compone del TGT, un timestamp y la propia petición del TGS, cifrada con la clave de sesión que envió previamente el AS.

En caso de que todo el proceso sea exitoso, el *Ticket Granting Server* generará una nueva clave de sesión y enviará al usuario el TGS.

El paquete en que envía el *Ticket Granting Server* el TGS al usuario, va cifrado con la clave de sesión que previamente generó el AS, y que solo el AS y el usuario conocen. El TGS recibido por el usuario contiene: una copia de la **nueva clave de sesión** generada por el *Ticket Granting Server*, un nuevo **ticket de sesión**, e información del servidor, así como un **timestamp**.

Con la clave de sesión enviada a través del TGS, y el servicio correctamente configurado para la autenticación remota por medio de Kerberos, el usuario puede autenticarse en dicho servicio empleando tan solo el TGS.

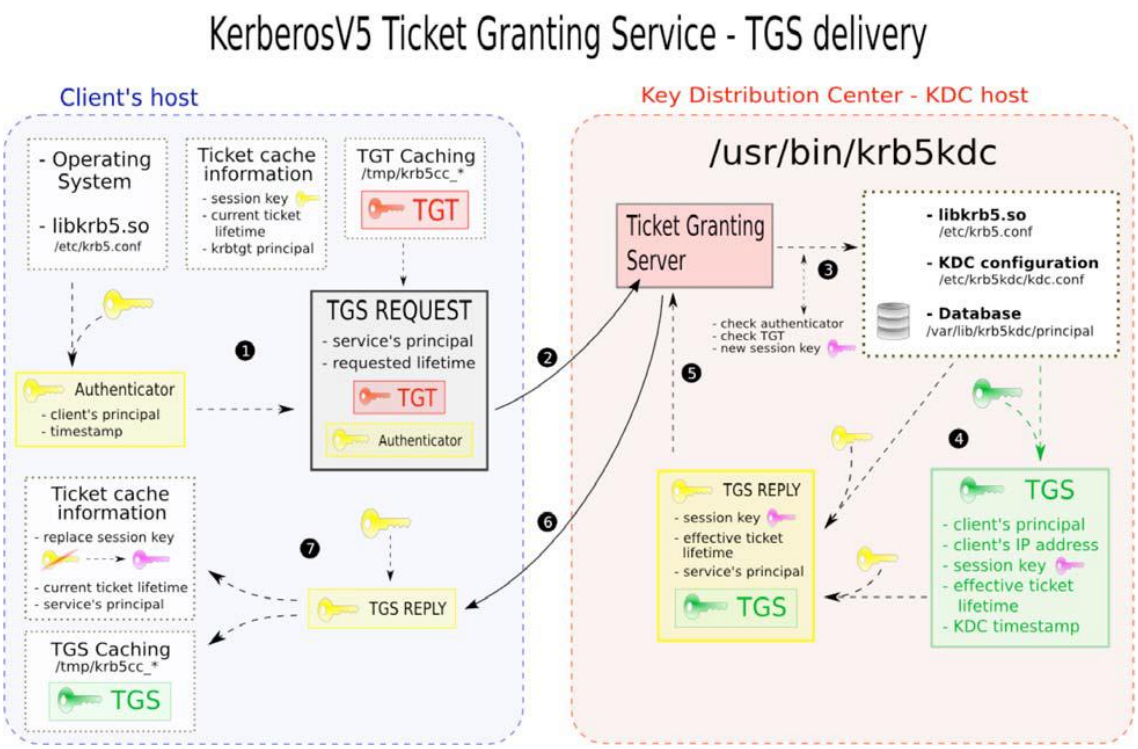

Imagen 8, descripción gráfica del proceso de obtención del TGS, obtenida de Wikipedia

En la *imagen 9* se puede observar un resumen de todo el protocolo que aquí se ha detallado. Para una mayor información acerca de su funcionamiento y configuración se puede acceder a la fuente detallada en la bibliografía de este trabajo [2].

Para concluir este apartado se debe comentar que, una **característica a tener en cuenta** que posee el empleo de este protocolo **de cara a la solución** que se pretende encontrar, no es otro que el **empleo de criptografía simétrica**. La solución que se pretende encontrar se prefiere que esté basada en el uso de certificados, es decir, criptografía asimétrica, de forma que **no solo no se deba reintroducir la password en cada acceso**, sino que **la solución debe ser carente de la misma** o en cualquier caso emplearla como sistema de protección añadido.

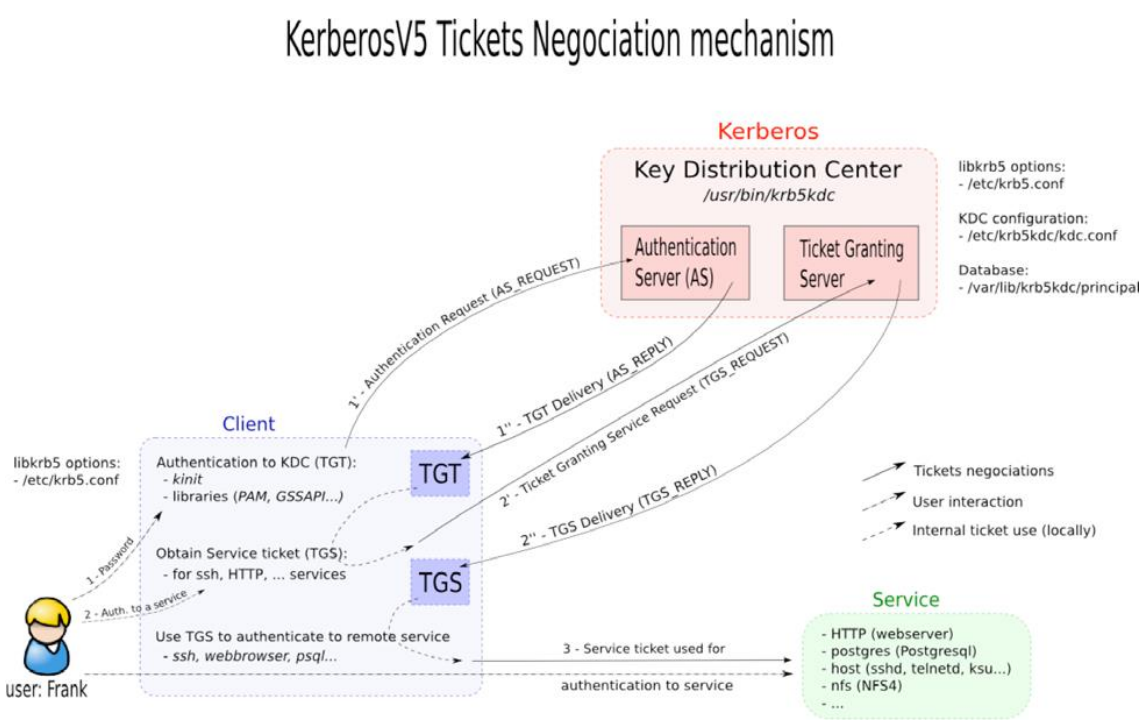

Imagen 9, resumen gráfico del funcionamiento del protocolo Kerberos, obtenido de Wikipedia

2.6. Sistemas biométricos para acreditación de acceso a usuarios

A pesar de que actualmente se escuchan numerosos avances en el campo de los sistemas biométricos [4], éstos aún se encuentran en un estado muy prematuro. Esto quiere decir que para autenticar a una persona mediante un sistema biométrico, el sistema no se puede considerar suficientemente sólido aún; además, al no estar fuertemente implantados, son sistemas mucho más costosos que el resto.

Por estos motivos se ha decidido, junto con los miembros del CeSViMa, que son sistemas que no deben ser tenidos en cuenta de cara a la solución que se debe diseñar para el centro. Sin embargo, dado que la principal finalidad de este proyecto es el estudio de los sistemas de autenticación alternativos a las clásicas passwords, se va a explicar en la medida de lo posible qué es un sistema biométrico, ejemplos de sistemas biométricos, y los problemas que aún plantean.

Qué es un sistema biométrico

Siguiendo la búsqueda que el ser humano siempre ha llevado de cara a encontrar la forma de identificar de forma única a un individuo, se llegó a emplear los llamados sistemas biométricos.

Los **sistemas biométricos** son sistemas que **tratan de identificar individuos mediante** características o *fingerprints* únicos en cada individuo, es decir, buscan **características físicas o de comportamiento que hagan único a cada usuario en el mundo**.

Con los sistemas biométricos se tendría un sistema de autenticación e identificación infalible, preciso, y altamente práctico. Pues se basan en características

del propio individuo, luego se **evita el hecho de tener que memorizar passwords, o incluso tener que portar algún tipo de dispositivo o tarjeta** a la hora de autenticarse en un sistema.

La forma en que **se mide la eficiencia** de estos sistemas son los ***falsos positivos*** y los ***falsos negativos***. No sirve el hecho de que un sistema sea imposible de falsificar, tampoco debe fallar en caso de que el *fingerprint* sea correcto, es decir, no debe tener un alto porcentaje de *falsos negativos*. De la misma forma el sistema de autenticación tampoco de permitir, bajo ningún concepto, casos en que por parecido que sea el *fingerprint* al original, se admitan como si fuesen el auténtico, es decir, el porcentaje de *falsos positivos* debe tender a cero.

Los **sistemas biométricos se apoyan**, como se ha mencionado, en **características biológicas que deben cumplir ciertos requisitos**:

- Debe ser capaz de **identificar de forma única** a un individuo del resto.
- La **falsificación** de dicha característica debe resultar **imposible**.
- **El paso del tiempo en el individuo no debe afectar al valor de dicha característica** desde el punto de vista del sistema biométrico.
- El sistema biométrico debe ser **viable en cuanto al coste computacional y monetario** en su desarrollo.
- El sistema biométrico **no debe resultar dañino a los usuarios** que lo empleen.

Estas características únicas, de las que se ha estado hablando durante el apartado, pueden ser de varios tipos, aunque **se podrían clasificar en dos grandes grupos**:

- **Características fisiológicas**: Son características, o medidas antropométricas únicas, incluidas en el cuerpo humano y que no dependen de un patrón de comportamiento del individuo en cuestión.
- **Características de comportamiento**: Son características basadas en comportamientos comunes del individuo, es decir, no requiere memorizar nada por parte del individuo dado que lo que se comprueba son patrones de comportamiento que el usuario hace de forma inconsciente. La medición de este tipo de características debe admitir cierto margen de error.

Tipos de sistemas biométricos

Se pueden encontrar muchos tipos de sistemas biométricos, aunque en este caso se van a analizar las **empleadas de forma más común**, pues es un trabajo de estudio y no de investigación.

Comenzando con las **características antropométricas**, es decir, las agrupadas entre las características fisiológicas, se pueden encontrar entre las distintas soluciones, las siguientes:

- **Características faciales**: Se toma como medida biométrica **la cara del individuo**. Se pueden encontrar las variantes de la **toma en dos dimensiones** o la **toma en tres dimensiones**, considerando más fiable ésta última.
  La eficiencia de este tipo de análisis aumenta en función de la potencia de cálculo que posea el ordenador que deba analizar la imagen. Esto se debe a que puede fraccionar la imagen en un número de celdas mayor en función de su capacidad de cómputo; se aumenta de esta forma la fiabilidad del sistema biométrico.

- **Análisis de retina**: Para analizar esta característica fisiológica se toma como medida **la vasculatura de la retina**, es decir, se toma como medida la imagen de los vasos sanguíneos que posee la retina del individuo. No solo es una característica única de cada usuario, sino que los órganos oculares se degradan a gran velocidad, luego dificulta la falsificación por extracción del órgano.

- **Análisis del iris**: El análisis del iris humano funciona de forma similar al análisis de retina; comprobándose **la vasculatura del iris** del individuo.
  El análisis de iris es considerado **más efectivo que el de retina**, ya que su proceso **consiste en tomar imágenes en blanco y negro del iris mientras se le aplica deformaciones mediante cambios lumínicos del ambiente**. Este tipo de análisis genera transformaciones matemáticas que resultan más complejas que las generadas con cualquier otro tipo de análisis biométrico.

- **Geometría de la mano**: Para este tipo de análisis **se evalúa la geometría de la mano al ser situada la misma sobre un lector, el cual analiza parámetros de longitud, grosor, y anchura en distintos puntos de la misma**. Suelen poseer un control de accesos en bases de datos para evaluar procesos de adelgazamiento o ensanchamiento de las medidas, ya que es fácil que cambien con el paso del tiempo.
  Estos sistemas son **de los más rápidos**, lo que hace que sean muy extendidos en su uso. Sin embargo son **de los que poseen una tasa de fallo más elevada**, pues es muy fácil que se produzcan *falsos positivos*; es común que en el margen de error que los sistemas permiten, haya más de una mano parecida.

- **Huellas digitales**: Consiste en el análisis de la normalización de las huellas digitales de un individuo. Durante el análisis no se evalúa la huella dactilar sin más, sino que el sistema consta de una serie de espejos que corrigen ángulos, normalizando la huella dactilar.
  Con la normalización de la huella se trata de recoger unos patrones que hacen única la huella de un individuo; patrones como los arcos, bucles, y remolinos de la huella son los evaluados. Una huella tiene entre 30 y 40 patrones, de los cuales se cree que no existen dos huellas que tengan más de ocho patrones idénticos.
  Al igual que los sistemas consistentes en el análisis de la geometría de la mano, son sistemas muy extendidos y rápidos de realizar.

Por otro lado, se tienen los **sistemas biométricos basados en patrones de comportamiento**, los cuales como ya se ha mencionado, evalúan el desarrollo de estos patrones para identificar de forma única a cada usuario.

- **Ritmo de escritura**: Se evalúa **el ritmo en que un usuario escribe sobre un teclado**, ya que se dice que es diferente entre cada individuo. Se suelen evaluar las pausas producidas al teclear ciertas palabras, el tiempo de presión, la forma en que se presionan las teclas, errores gramaticales…

- **Características de la voz**: De igual forma que se evalúa el ritmo de escritura, **se evalúa** en este caso **el ritmo de habla**. Para analizar los patrones de voz se evalúan desde los **tonos de voz** (graves y agudos), hasta las mismas pausas en el habla, **forma de pronunciación**, y demás características que hacen único a un individuo.
  El análisis de voz se debe realizar siempre en el mismo medio, pues no deben afectar factores de ruido, eco y reverberaciones.

- **Firma dinámica**: **La propia firma manual** que se ha empleado desde hace muchos años se puede considerar un factor biométrico. **Aunque no se emplea a la hora de autenticar a usuarios mediante un sistema automático**, es posible identificar individuos mediante la misma; se evalúan los tipos de trazos que realizan los usuarios a la hora de estampar su propia firma de forma manual.

Estado actual de los sistemas biométricos

Como se ha mencionado al comienzo de este apartado, estos sistemas a pesar de resultar aparentemente eficientes, pudiendo parecer la solución ideal a los problemas de autenticación de usuarios, han sido descartados de entre las posibles soluciones a definir en el centro CeSViMa.

Los sistemas biométricos presentan actualmente multitud de problemas que los hacen inviables, o ser obviados a la hora de servir de sistemas de autenticación.

En primer lugar, si se los analiza en función de los dos grandes grupos que se han definido previamente, **los sistemas biométricos basados en el análisis de patrones**

de comportamiento, son muy fáciles de engañar; los ritmos de escritura pueden ser capturados mediante ataques de ingeniería social, y los escáneres de voz mediante grabaciones. Además este tipo de sistemas biométricos poseen unos márgenes de error muy amplios para evitar *falsos negativos*, que los hacen a su vez **muy vulnerables a provocar *falsos positivos***. Y por último, se debe tener en cuenta que cambios tan comunes y naturales en la voz, como los causados por una enfermedad, invalidarían el análisis de voz provocando un *falso negativo*.

Por otro lado se tienen **los sistemas fisiológicos**, donde se tiene el problema de encontrar una característica fisiológica fiable, pues **no es nada sencillo encontrar una característica que permanezca invariable a lo largo de la vida del usuario**.

Características como la medición del ritmo cardíaco se comienzan a desarrollar, con proyectos como Nymi [17], o la medición de la geometría de la mano. Sin embargo, son sistemas que varían muchísimo en un período breve de tiempo, en el caso del ritmo cardíaco, o tienen un margen de error muy grande dando lugar a *falsos positivos*, como el caso de la geometría de la mano. Se debe tener en cuenta que los *falsos positivos* nunca se deben producir en un sistema de autenticación sólido.

En el caso de características como el escáner facial, o la huella dactilar, son sistemas biométricos que se suelen emplear como medidas de protección estándar en un sistema de autenticación; su coste es muy reducido, al igual que la dificultad de su falsificación. Falsificar estos sistemas es tan sencillo como mostrar una foto del individuo o un molde de la característica; en el caso de la huella es un sistema sencillo de falsificar, pues se van dejando en todas las superficies y basta un molde de silicona para engañar a un sistema biométrico basado en escáneres de huellas dactilares. Un claro ejemplo de sistema de autenticación débil por huella dactilar, es el empleado por el dispositivo IPhone 5S de Apple, pudiéndose encontrar tutoriales y vídeos mostrando el proceso.

Los sistemas basados en escáneres faciales o huellas dactilares tienen además el mismo problema que los anteriores. Son características que es fácil que cambien con el paso del tiempo, pues el hecho del crecimiento de barba, o una herida en el dedo en que se ha recogido la huella dactilar, son suficientes para que falle la autenticación, generándose de esta forma un *falso negativo*.

Los únicos sistemas que se pueden considerar realmente útiles, invariables e infalibles, de todos los vistos a la hora de tratar los sistemas biométricos, son los basados en escáneres vasculares. Ejemplos de estos son los escáneres oculares de retina e iris. Este tipo de sistema biométrico no es nada sencillo de falsificar, pudiéndose considerar casi imposible, además no varían con el paso del tiempo. **Sin embargo** presentan un gran problema, **ese tipo de escáneres basados en radiaciones con luz infrarroja es muy costoso**, hasta el punto de cuestionarse si su necesidad es requerida por el sistema que se pretende proteger.

Como **conclusión del estado** en que se encuentran los sistemas biométricos, **en opinión del autor de este documento**, se puede considerar que **son un gran complemento junto al resto de sistemas de autenticación**; se puede ver en el apartado *2.7. Sistemas de control de acceso mediante Smart Cards*, que en el DNI-e de España se incluye información de la huella dactilar, la fotografía, y firma del

usuario, fortaleciendo así un sistema de seguridad basado en criptografía asimétrica con características biométricas. **Sin embargo, distan mucho de ser sistemas de autenticación sólidos por sí mismos, a excepción de los escáneres vasculares, donde en caso de querer proteger un servicio mediante un sistema de autenticación, sin tener en cuenta costes, son la mejor solución hoy en día**.

2.7. Sistemas de control de acceso mediante Smart Cards

En este apartado se procederá a explicar algunos de los procedimientos e implantaciones de las Smart Cards en los sistemas de autenticación, enfocando el apartado de cara a la solución que interesa para el CeSViMa.

En primera instancia se definirán y darán a conocer conceptos básicos acerca de las Smart Cards, sus utilidades, y modos de operación. De esta forma se pretende introducir al lector el conocimiento necesario para la comprensión, y explicación, del por qué se quiere tender actualmente a asegurar los sistemas de autenticación con estos dispositivos.

¿Qué son las Smart Cards?

El concepto de Smart Card o "tarjeta inteligente", engloba a todas las tarjetas que poseen circuitos integrados y que permiten la ejecución de cierta lógica programada. El **tamaño** más característico de estas tarjetas es el **ID-1**, el cual se corresponde al de cualquier carnet o tarjeta de bolsillo; también se suelen encontrar con tamaño **ID-000**, siendo éste el modelo de las conocidas tarjetas SIM, empleadas para los teléfonos móviles.

Las **Smart Cards** poseen componentes que **permiten la transmisión, el almacenaje y el procesado de datos**. Los datos pueden ser transmitidos empleando lectores de contacto, como pueden ser las tarjetas de crédito o DNI-e, o mediante sistemas de campos electromagnéticos empleando la tecnología NFC (Near Field Communication).

Las principales ventajas de las Smart Cards vienen a ser las siguientes:

- **No requieren de alimentación** para los sistemas de procesamiento y memorias que poseen, ya que se la proporcionan los propios lectores de tarjetas.

- Poseen sistemas de seguridad que **no permiten que los datos almacenados sean accedidos de forma externa**. Estos sistemas de protección evitan la lectura externa e indebida de datos, así como en el caso de las tarjetas criptográficas evitan la extracción indebida del certificado o clave privada del usuario. Estos sistemas, si bien no son 100% infalibles, dificultan en gran medida el acceso a todo lo almacenado en las Smart Cards.

- Las Smart Cards más actuales, **ofrecen el almacenaje y administración de datos para soportar varios sistemas en los que autenticar a un usuario.**

- **La gran comodidad y portabilidad** que ofrecen, pues su tamaño es el de una tarjeta o carnet de bolsillo.

Tipos de Smart Cards

Se puede clasificar las Smart Cards en dos grandes grupos en función de las capacidades de procesamiento que posean.

- Se puede distinguir las **tarjetas de memoria**, las cuales son empleadas como simples contenedores de ficheros o datos; carecen de procesador, como se puede ver en la *imagen 10*.
  El único sistema de seguridad que poseen estas tarjetas es la protección de escritura o borrado de determinadas regiones de memoria, y en caso de incorporar algoritmos criptográficos éstos son muy sencillos. Suelen estar implementados en los propios circuitos de las tarjetas. Un ejemplo de éstas son los primeros modelos de tarjetas *MIFARE*, que a pesar de emplearse como método de autenticación en algunos servicios, tan solo poseían algoritmos de cifrado sencillos como es el *DES*.

- Por otro lado están las **tarjetas con microprocesador**, las cuales poseen una arquitectura análoga a la de un computador; posee microprocesador; memoria RAM para almacenar de forma volátil los datos tratados por la CPU; y la memoria ROM, donde se almacena “de fábrica” la información para el sistema operativo que emplea la tarjeta.
  Los datos y códigos de programas a ser interpretados son almacenados en la EEPROM, que es un chip de memoria no volátil. Un ejemplo de datos o información almacenada en la EEPROM podría ser el certificado de un usuario, en el caso de emplearse dicha tarjeta como sistema de autenticación con criptografía asimétrica.
  En este tipo de tarjetas es posible implementar algoritmos criptográficos más complejos, pues al poseer un microprocesador la tarjeta ya puede realizar operaciones matemáticas más complejas.
  A pesar de todo siempre se debe tener en cuenta el consumo eléctrico que generan, pues la alimentación suele depender de la parte de los lectores de tarjetas, luego se debe buscar un consumo ínfimo.

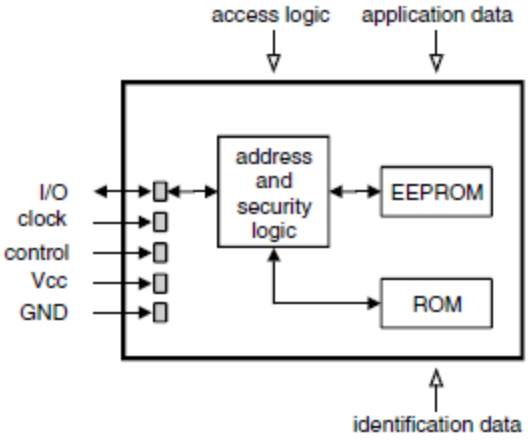

Imagen 10, arquitectura típica de una tarjeta de memoria.

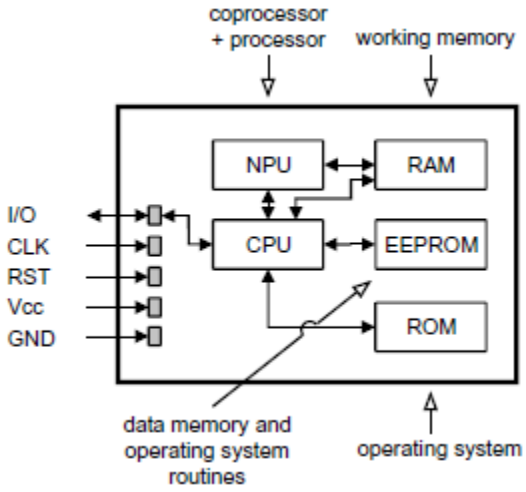

Imagen 11, arquitectura típica de una Smart Card con procesador.

Casos de uso de las Smart Cards

Los sistemas en que se aplica el uso de Smart Cards, al igual que los Token USB como se explica en el apartado *2.8. Sistemas de control de acceso mediante Token USB*, son distribuidos por numerosas empresas dedicadas a la venta de servicios de protección y autenticación de sistemas. Dichas empresas emplean estos dispositivos para crear lo que se conoce como *Two factor authentication*; gracias a las Smart Cards y Token USB, disponen de un almacén de claves y certificados para criptografía asimétrica, pero a su vez protegen éstos con una clave de acceso a las propias Smart Cards y Token USB.

Sin entrar en detalles ni publicitar a ninguna de dichas empresas, se nombrarán en este apartado los más conocidos o empleados en España, bien por la iniciativa que pretenden promover, o bien por la fiabilidad que la empresa distribuidora otorga.

Por un lado se encuentra el **DI-e**, las nuevas versiones del Documento de Identidad en distintos países del mundo, el cual es emitido por una autoridad oficial

para permitir la identificación de la población de forma personal y/o virtual. España es uno de los países que más extendido tiene el uso de este dispositivo con su **DNI-e**, habiéndolo incorporado entre su población desde el año 2006.

Las principales **ventajas** del **DNI-e**, y los usos que se pretenden conseguir en un futuro, ya que actualmente no se permite su uso en prácticamente ningún servicio oficial del Estado, son las siguientes:

- **Aumento de seguridad física del DNI**, pues posee una mayor cantidad de medidas para evitar su falsificación, reduciendo así el número de casos de suplantación y fraude.

- **Autorización y acceso a los distintos servicios** (estatales o privados) las 24 horas del día, todos los días del año, y desde cualquier parte del mundo. Esta característica, si bien es cierto que las páginas del Estado que requieren autenticación permiten el uso del DNI-e, no se emplea actualmente de forma rigurosa y obligatoria.

- **Ejercer el voto electrónico**. Si bien ésta sería una característica útil a otorgar al DNI-e, aún hay muchos aspectos a solucionar en cuanto a que sea un mecanismo 100% fiable y capaz de emular al hecho de "votar en persona".

- **Firma de documentos y pago de impuestos en línea**. Ésta sería una de las características más útiles, pues significaría la total implantación de la criptografía asimétrica mediante Smart Cards. Sin embargo, actualmente aún requiere de un tiempo para asegurar su uso e implantación de forma tan abierta como para que lo emplee todo un país.

El **DNI-e** es un más que correcto ejemplo de diseño e implantación de Smart Card en España. Posee un chip capaz de almacenar de forma segura la misma información que se encuentra en la tarjeta de forma impresa; además **almacena** imágenes digitalizadas de la **fotografía, firma manuscrita, impresiones dactilares**, **y** los **certificados digitales** necesarios para realizar la autenticación y firma electrónica.

Por otro lado se debe comentar, el servicio proporcionado a nivel nacional por la ***Fábrica Nacional de Moneda y Timbre (FNMT)*** [18], quien entre sus muchos servicios se encarga de expedir certificados para una correcta autenticación en sistemas *Two Factor Authentication*.

La FNMT es la entidad certificadora de España, por lo que todos los certificados expedidos y firmados por la misma son considerados válidos por cualquier servicio nacional. Proporcionan un servicio de ventas de tarjetas criptográficas desde su sitio web, donde entre los muchos tipos de tarjetas se encuentran las dedicadas a *identificación electrónica*. En su página web la propia FNMT declara que:

"Las técnicas criptográficas utilizadas en las tarjetas de certificación digital de la FNMT-RCM, permiten a la Administración, al ciudadano y a la empresa realizar sus trámites y transacciones electrónicas a través de la red con garantías de máxima seguridad".

El soporte empleado por CERES (iniciativa puesta en marcha por la Administración de España, liderando la FNMT) posibilitando el uso de los certificados de la FNMT, proporciona técnicas criptográficas como el algoritmo de cifrado simétrico Triple-DES, el algoritmo de cifrado asimétrico RSA, con manejo de claves de hasta 2048 bits, así como la generación de funciones unidireccionales hash mediante los algoritmos SHA-1.

Con todo esto, y como **conclusión** de este apartado, se puede comprobar cómo actualmente **se tiende al desarrollo e implantación de la firma electrónica y los certificados digitales**, pues es la única forma de asegurar la identidad de una persona en un mundo que se basa en Internet. A pesar de que la **iniciativa nacional con el DNI-e** es **muy buena**, hace **falta un mayor apoyo y desarrollo** de la misma. Fomentar el uso del DNI-e sería un gran paso hacia su instauración en la autenticación digital.

2.8. Sistemas de control de acceso mediante Token USB

Los sistemas de control de acceso mediante Token USB son muy similares a los ya vistos en el apartado *2.7. Sistemas de control de acceso mediante Smart Cards*. Aunque a diferencia de éstas, los Token USB no son empleados solo como dispositivo de almacenaje de certificados, formando una infraestructura típica PKI, sino que también son empleados como sistemas de autenticación *One Time Password* (OTP).

En los siguientes sub-apartados se definirá qué son exactamente los Token USB, distinguiéndose los distintos tipos y generalizando entre la gran oferta que hay en el mercado actual. Se definirán también los casos de uso más comunes en los que se suelen introducir estos dispositivos.

¿Qué son los Token USB?

Son dispositivos que poseen un **hardware similar al de las Smart Cards con microprocesador**, es decir, poseen una estructura similar a la que se puede ver en la *imagen 11*.

Los Token USB **son dispositivos capaces de procesar mediante su hardware, el cifrado de datos, correos electrónicos, acceder a servicios y redes privadas de forma segura, y dar cabida al uso de la firma digital**.

El hardware que poseen permite el **uso** tanto **de criptografía simétrica**, con los algoritmos considerados actualmente como válidos (AES, RC4, SHA-256); como también permiten el uso de **criptografía asimétrica** (RSA, Diffie-Hellman), pues almacenan de forma segura certificados y claves privadas de usuarios.

Su diferencia respecto a las Smart Cards, radica en que **son dispositivos del tipo PenDrive**, luego su alimentación proviene comúnmente del computador al que se conectan vía USB. Una excepción son los Token USB del tipo *One Time Password*, los cuales son explicados más a fondo en el apartado *4. Informe de vigilancia sobre soluciones propuestas en el mercado actual*, que poseen un display y pueden ser alimentados por pilas internas.

Estos dispositivos son **distribuidos por numerosas empresas**, entre las que destacan **RSA** con sus soluciones OTP, y diversas empresas como **Kalysis** con sistemas más polivalentes pero enfocados al uso de certificados. Se ha destacado esta última entre las muchas existentes por ser una empresa Española en colaboración con el COIT, poseyendo múltiples certificados internacionales en sus productos.

Tipos de Token USB

Entre los tipos de Token USB, a parte de las diferentes características que puedan tener en función de la empresa que los distribuya, se pueden destacar dos grandes tipos:

- **Tokens One Time Password (OTP)**: Son dispositivos que, o bien se alimentan mediante una conexión USB al ordenador, o bien se alimentan mediante una pila interna, no requiriendo conexión USB; siendo esta opción la más común.
  Los tokens OTP disponen de un display en el que muestran al usuario una clave que deberá introducir en el sistema de autenticación, bien de forma independiente, o bien en conjunto a una password del propio usuario. Estas passwords que generan los tokens OTP, tienen una validez muy breve, pues pueden caducar en cortos períodos de tiempo, o tras cada autenticación.
  Los tokens OTP **se encuentran sincronizados con el servidor del sistema de autenticación, sabiendo el mismo la clave válida en cada momento**. Para una explicación más detallada se puede consultar el sub-apartado *Sistemas One Time Password (OTP)* del apartado cuatro de este documento.
  Un ejemplo visual del tipo de token con alimentación a pila, es decir, sin conexión al terminal del usuario, se puede ver en la *imagen 13*.

- **Token USB**: Se ha definido el otro conjunto como token USB, siendo éste un nombre muy genérico, haciendo referencia a los dispositivos que carecen de display y que, sobre todo, no son sistemas OTP.
  Son dispositivos como los distribuidos por la empresa Kalysis, cuyo **aspecto** es el **de un simple PenDrive**. En su interior albergan un **hardware similar al que poseen las Smart Cards**, dándoles la funcionalidad de realizar operaciones criptográficas a nivel de hardware, así como almacenar certificados y claves privadas sin permitir que éstos salgan del dispositivo sin a previa introducción del PIN de desbloqueo.
  Suelen ser compatibles con estándares PKCS#11 y PKCS#15, al igual que las Smart Cards, permitiéndoles en la mayoría de los casos ser **compatibles con múltiples plataformas software a nivel de sistemas operativos**.

Casos de uso de los Token USB

Los Token USB son sistemas muy empleados actualmente por empresas para mejorar, y asegurar, los sistemas de autenticación de sus servicios.

Son **sistemas muy fiables**, pues permiten almacenar de forma segura, considerada como casi inviolable, los certificados digitales y claves privadas almacenadas en su interior. Además, son dispositivos que se emplean para la firma electrónica entre sistemas informáticos, **permitiendo garantizar la autenticidad y el no repudio** del emisor de un mensaje o el usuario que realiza una transacción.

Dadas las similitudes que poseen respecto a las Smart Cards, la conclusión que el autor de este documento ofrece es la de que los dispositivos Token USB son, actualmente, uno de los mejores sistemas de autenticación por todas las posibilidades y facilidades que ofrecen. **En opinión del autor, poseen la ventaja respecto a las Smart Cards de que carecen de la adquisición o empleo de sistemas de lectura, pues poseen conexión USB**, que actualmente es **un estándar en todos los dispositivos informáticos**. De esta forma son más versátiles, mejorando aún más si caben las ventajas de las Smart Cards.

3. Estudio de la solución actual en CeSViMa

El Centro de Supercomputación y Visualización de Madrid (CeSViMa), fue creado a finales del año 2004. Está dedicado al almacenamiento masivo de información, computación de altas prestaciones, y visualización interactiva avanzada; es uno de los sistemas o infraestructuras más importantes de la Universidad Politécnica de Madrid (UPM).

En su interior alberga el segundo nodo más importante de la Red Española de Supercomputación, el cual está compuesto por 1.036 nodos eServer BladeCenter JS20, cada uno de los cuales dispone de 2 procesadores PPC de 2'2 GHz (8.8 GFlops) con 4 GB de RAM, así como 168 nodos eServer BladeCenter JS21con 4 procesadores PPC 2'3GHz (9.2 GFlops) con 8GB de RAM [25]. El Magerit, nombre del supercomputador que alberga el CeSViMa, puede proporcionar una potencia de cálculo de hasta 103.5 TFlop/s.

Durante el desarrollo de este trabajo se han evaluado los distintos sistemas de autenticación, llevados a cabo en distintas empresas en sus respectivos servicios y sistemas. Sin embargo, siempre se ha tratado de enfocar el problema y las posibles soluciones de cara esta infraestructura de la UPM.

En este apartado se detallará el sistema que actualmente protege y separa a posibles atacantes del acceso al supercomputador Magerit; pues el acceso que se pretende tratar no es otro que el de los usuarios y/o administradores que trabajan en el propio centro (definidos durante este apartado como *usuarios*), es decir, de aquellas personas que en sus credenciales de acceso tienen total permiso para emplear todas las características del supercomputador Magerit.

Actualmente, el **sistema de acceso** que separa a los *usuarios* del supercomputador Magerit (*core*), consiste en un **acceso mediante el protocolo SSH a un servidor** que se denominará como ***gateway*** o entrada al core. Dicho *gateway* es el **que proporciona acceso a los *usuarios* hasta el *core***, es decir, es el **único punto de entrada** al *core* del supercomputador, lo que facilita la labor en la protección del acceso al núcleo del sistema.

El *gateway* posee un sistema operativo Linux, lo que da a entender que la **configuración a su acceso**, o preferencias de conexión al servidor, se pueden definir **mediante archivos de configuración como puede ser el PAM** (Pluggable Authentication Modules). Gracias al archivo de configuración PAM es posible unificar los distintos módulos de autenticación de los distintos programas que posea un servidor.

Por otro lado, **los *usuarios* hacen uso de distintos sistemas operativos**, como son Windows, MacOS y distintas distribuciones Linux. Este hecho **dificulta en gran medida la búsqueda de una solución única y compatible** para todas las versiones, ya que los distintos S.O. difieren mucho en lo que a su estructura se refiere.

En el entorno ya definido, tal y como se puede ver en la *imagen 12*, **los *usuarios* tan solo deben acceder mediante una conexión SSH al *gateway,*** empleando cada uno de ellos distintos clientes o agentes para gestionar sus claves, **y donde a través**

del mismo se accede al *core*. Dicho esto, se puede simplificar comentando que **lo único que separa a los *usuarios* del propio *core*, tan solo es la password con la que crean el canal seguro SSH**.

La actual solución que implementa el CeSViMa presenta los problemas ya mencionados en el apartado *2.2. Autenticación mediante passwords y problemas en su uso*. Destacan entre ellos el hecho de que las passwords puedan ser captadas por terceras personas, y que a la hora de eliminar un usuario se deba comprobar y evaluar en cada uno de los sistemas del centro si se le ha revocado el acceso.

El objetivo de este proyecto, a parte del estudio y evaluación de las técnicas y métodos de acceso alternativos a passwords, que ya se han visto en apartados previos, no es otro que el **encontrar una solución que se adapte a las exigencias dadas por el CeSViMa**. Los principales puntos a tratar, o que debe cumplir la solución a encontrar, son los siguientes:

- **El sistema debe ser único para todos los servicios**, es decir, que no sea necesaria la comprobación de todos y cada uno de los servicios una vez que se requiera la revocación del acceso de un usuario al centro.

- **El sistema debe ser multiplataforma**. Dado que todos los *usuarios* poseen distintos S.O. la solución debería ser la misma, y poseer la misma configuración, independientemente del S.O. que posea el *usuario*.

- **El sistema debe carecer de password**, luego el establecimiento del canal seguro SSH debe formarse sin necesidad de la introducción de una password por parte del *usuario*. En caso de requerir una password, tan solo debe ser para asegurar aún más el acceso, es decir, debe ser una medida adicional de seguridad.

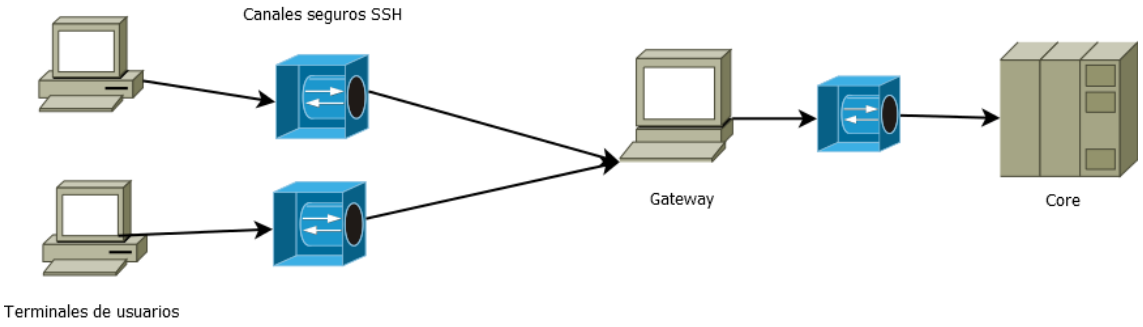

Imagen 12, diagrama de la situación actual

4. Informe de vigilancia sobre soluciones propuestas en el mercado actual

A fin de incorporar una solución idónea para el centro CeSViMa, se ha realizado un estudio para evaluar las soluciones llevadas a cabo en el mismo ámbito por otras empresas. Se ha recopilado información al respecto de empresas que proporcionan como servicio la incorporación, instalación, y mantenimiento de sistemas de autenticación, tal y como se ha ido mencionando a lo largo de la memoria.

El estudio se ha dividido en dos vertientes, por un lado se ha focalizado en comprobar las soluciones que proporcionan las distintas empresas encargadas de instaurar un sistema de autenticación fuerte; empresas conocidas como RSA, Safenet, o SecuTech, o incluso las soluciones proporcionadas por el gigante Google. Por otro lado, se ha intentado obtener información acerca de las soluciones adoptadas por empresas reales de cara a la autenticación de sus usuarios en sus sistemas.

Tras esto se han podido agrupar las distintas soluciones en tres grupos desde el punto de vista del autor de este trabajo. Los distintos grupos definidos se detallan a continuación.

Sistemas One Time Password (OTP)

Los sistemas One Time Password son capaces de **generar una clave de acceso considerada de un nivel de seguridad alto**. La diferencia con los sistemas de autenticación basados en passwords radica en que, los OTP generan contraseñas **nuevas en función del tiempo o del uso de las mismas**.

Un sistema OTP mantiene una sincronización entre cliente y servidor, de forma que ambos conocen la password considerada como válida en cada momento. La actualización de la clave puede variar, siendo comúnmente el tiempo el determinante a la hora de caducar las passwords, aunque también se pueden configurar para que sea cada acceso el que produzca la caducidad de la misma.

Estos sistemas son capaces de proporcionar un acceso seguro a infraestructuras, así como generar canales seguros por SSH; pues tan solo requieren de un cliente o agente capaz de tratar los distintos cambios de las passwords, que además debe cuadrar en el ámbito de la empresa que lo utilice o requiera.

Hay bastantes **empresas que distribuyen estos sistemas OTP**, entre las que **destaca RSA**. La mayoría de estas empresas ofrecen dispositivos del tamaño de un **USB**, donde por medio de un **display** informa al usuario del token que debe agregar a su password para el acceso al sistema; en la *imagen 13* se muestra un ejemplo del dispositivo, conocido como SecurID. Se debe tener en cuenta que el emplear estas soluciones nunca garantizan una seguridad total, pues RSA ya ha tenido serios problemas con sus dispositivos SecurID; son conocidos el ataque sufrido por dos grupos de cibercriminales [26] en el 2011, lo que les obligó a modificar el sistema SecurID, y el reciente descubrimiento de que la NSA posee puertas traseras en el algoritmo que actualmente poseen los dispositivos SecurID [27]. La propia RSA desaconseja el uso de sus dispositivos hasta que el NIST resuelva los problemas,

aunque algunas empresas han realizado modificaciones en las normas de uso de los mismos, como concatenar las passwords de los usuarios al número generado por el SecurID.

Otro caso especial es el del gigante **Google**, quien está realizando un sistema propio para mejorar su entorno mediante OTP. El llamado ***"Google Authenticator"*** difiere de los sistemas antes mencionados, ya que emplea los teléfonos móviles de los usuarios como dispositivos OTP. Además tiene disponible el código fuente para su implementación y colaboración en el desarrollo.

Imagen 13, Dispositivo RSA SecurID

Las soluciones de este estilo se suelen enfocar a modo de sistema de autenticación único, donde la autenticación consiste en la introducción del token dado por el dispositivo, o bien una mezcla del token y una password del usuario. Sin embargo también hay otras, como en el caso de Google, en las que el código generado por el sistema OTP se suma a la introducción de la password clásica, fortaleciendo así el sistema de autenticación; también hay otras empresas que para que el propio dispositivo OTP genere una password, se debe introducir una clave de acceso, es decir, los dispositivos se encuentran protegidos de cara a terceras personas.

Sistemas Two Factor Authentication basados en PKI

Los sistemas Two Factor Authentication ya han sido explicados a lo largo de este trabajo. Son sistemas de autenticación de usuarios, cuya autenticación **consiste en que el usuario presente algo que él mismo conoce**, como es una password, **y algo que posee en común con el servidor**, como puede ser un certificado o un token. Un claro ejemplo es el sistema *Google Authenticator* descrito en el sub-apartado anterior.

En este sub-apartado se pretende centrar este método de autenticación en el caso en que se quiere emular una **infraestructura PKI**; esto es, **cuando la parte que cliente y servidor "tienen en común" es un certificado, o una clave previamente generada (pares de clave pública y privada)**.

Sin entrar en detalles de cómo funciona y se forma una infraestructura de clave pública (PKI), pues para ello se puede consultar el apartado *2.3.3. Infraestructura de clave pública*, se procederá a hablar de las soluciones dadas por las distintas empresas que se han consultado.

Las soluciones que distribuyen las distintas empresas, consisten en el uso de las Smart Cards o Token USB para el almacenaje y control de los certificados o claves del usuario. Aunque hay excepciones, donde las empresas ofrecen además la instalación completa de servidores y equipos para administrar de forma completa la infraestructura PKI; naturalmente son las opciones más costosas, pues ofrecen un completo mantenimiento. Para una mayor información acerca de las Smart Cards o los Token USB, se pueden consultar los apartados *2.7. Sistemas de control de acceso mediante Smart Cards* y *2.8. Sistemas de control de acceso mediante Token USB* respectivamente.

Soluciones adoptadas en empresas reales

En este último sub-apartado se pretende mostrar las soluciones adoptadas realmente por empresas actualmente. Para ello se ha podido tener acceso a información de distintas empresas que, por motivos obvios, no serán nombradas en este trabajo.

Básicamente, los sistemas de autenticación empleados actualmente por las empresas, distan mucho de ser seguros. Tal y como se mencionó en la introducción, la mayoría de ellas optan por el simple uso de passwords para el acceso a sus servicios y servidores, lo que acarrea todos los problemas que ya se mencionaron en el apartado *2.2. Autenticación mediante passwords y problemas en su uso*.

Aunque gracias a que hoy en día se están haciendo públicos todos los ataques y fallas de seguridad en los distintos sistemas y servicios de populares empresas, así como las enormes pérdidas económicas que ellas suponen, la situación comienza a cambiar. Muchas empresas optan por mejorar sus sistemas de autenticación, siendo los métodos nombrados anteriormente los más populares.

El principal problema que aún siguen poseyendo las empresas es, que a pesar de emplear mejores sistemas de seguridad a la hora de acceder a sus sistemas, siguen sin proteger la mayor parte de sus accesos. Con esto se quiere decir que, no solo es necesario cambiar el uso de passwords por el de certificados o claves públicas y privadas, sino el conseguir que los equipos y medios empleados en **estos sistemas de autenticación sean empleados tan solo para uso de trabajo.**

Gracias a que se ha podido obtener información acerca de empresas que manipulan datos altamente importantes, se ha podido comprobar cómo protegen el acceso a sus sistemas. Éstas empresas no solo emplean sistemas como los mencionados previamente, sino que tienen equipos proporcionados por la empresa únicamente para que sean utilizados para propósitos relacionados con la misma. Esto no solo quiere decir que **tengan la mayoría de los servicios comunes de navegadores y servicios de correos "capados"**, sino que además **los equipos poseen fingerprints registrados por la propia empresa**; los llamados **equipos "masterizados"**. De esta forma, si un atacante quisiese acceder de forma remota a servicios de dichas empresas debería: en primer lugar hacerse con un equipo "masterizado" por la empresa, cuyo acceso sería denegado tan pronto se declarase como perdido o robado; en segundo lugar, poseer el dispositivo relacionado con la autenticación tal y como se vio en los apartados anteriores, que también podrían ser dados de baja por la empresa; y por último, conocer la password de acceso que tenga el usuario para conformar el sistema Two Factor Authentication.

Con todo esto se puede ver que, cuando a las empresas les interesa, es posible dificultar en gran medida el acceso de terceras personas, aunque ello requiere de una fuerte inversión y de accesos altamente incómodos a los sistemas y servicios de cara a los usuarios normales.

5. Diseño de solución

En este apartado se tratará de exponer las distintas soluciones, descritas por orden de preferencia, que se han estudiado en conjunto con miembros del CeSViMa. Los aspectos que se han tenido en cuenta, al igual que la situación y problemas que presenta el centro actualmente, se encuentran detallados en el apartado *3. Estudio de la solución actual en CeSViMa*.

Recordando los puntos a tener en cuenta en la búsqueda y posible implementación de la solución, se tienen los siguientes:

- **El sistema debe ser único para todos los servicios**, es decir, que no sea necesaria la comprobación de todos y cada uno de los servicios una vez que se requiera la revocación del acceso de un usuario al centro.

- **El sistema debe ser multiplataforma**. Dado que todos los *usuarios* poseen distintos S.O. la solución debería ser la misma, y poseer la misma configuración, independientemente del S.O. que posea el *usuario*.

- **El sistema debe carecer de password**, luego el establecimiento del canal seguro SSH debe formarse sin necesidad de la introducción de una password por parte del *usuario*. En caso de requerir una password, tan solo debe ser para asegurar aún más el acceso, es decir, debe ser una medida adicional de seguridad.

Con estos aspectos a tener en cuenta, las soluciones que se han estudiado durante el apartado *2. Análisis de técnicas de acreditación de acceso*, y el estudio de las soluciones empleadas actualmente en el apartado *4. Informe de vigilancia sobre soluciones propuestas en el mercado actual*, se detallarán a continuación las soluciones que se van a evaluar para una posible posterior implantación; así como los inconvenientes que las mismas posean.

Primera solución

La primera solución a tener en cuenta son los ya conocidos como **sistemas *two factor authentication***, los cuales, como ya se ha ido comentando a lo largo del trabajo, se basan en realizar la autenticación mediante algo que el usuario conoce (una password), y algo que el usuario posee en común con el servidor o que lo autentique se cara al mismo (un certificado o token).

Este tipo de solución es **distribuido por numerosas empresas** dedicadas a la seguridad de la información, por lo que son soluciones en mayor o menor medida costosas. Esta solución garantiza la satisfacción de dos de los tres puntos pedidos, pues es **capaz de proporcionar una solución unificada en cuanto a servicios**, además de ser un **sistema en que se fortalece la autenticación con el uso de passwords, sin ser el único método de acceso**.

El **principal problema** que se da en esta primera solución es el hecho de cubrir el segundo punto pedido, dado que **no todos estos sistemas garantizan**

compatibilidad multiplataforma, dado que no es sencillo encontrar un agente SSH común entre ellas. Además surge otro problema, y es que **no todas las soluciones two factor authentication soportan el uso de certificados X.509**.

A pesar de los problemas, dado que el estándar X.509 es el que está comenzando a predominar actualmente, se empiezan a encontrar soluciones que satisfacen este problema, por lo que se deberá indagar entre las muchas disponibles.

Segunda solución

La segunda solución que surge es el empleo de sistemas *One Time Password*, que como también se ha ido mencionando a lo largo del trabajo, consiste en que el servidor pida una clave al usuario, aunque dicha clave se actualiza con cada uso o según el momento de tiempo en que se vaya a emplear. Con esto se tiene que solo el servidor y el usuario conocen como se ha ido derivando dicha clave, es decir, a pesar de ser una solución basada en una password no es vulnerable a keylogers, pishing y demás ataques realizados a sistemas de autenticación mediante passwords.

Se debe tener en cuenta que el uso de OTP se puede conjuntar con otros sistemas de autenticación, formando así una solución two factor authentication.

Al igual que la primera solución, esta segunda satisface el hecho de la ausencia de la introducción de passwords por parte del usuario. Sin embargo, también presenta los mismos problemas, pues es complejo encontrar una solución en que haya un agente común a todos los sistemas operativos, y que además sea capaz de ofrecer la autenticación OTP a todos los servicios y aplicaciones que quieran acceder al supercomputador.

En este tipo de soluciones se ha encontrado además otro problema añadido, y es que la mayoría de ellos basan sus mecanismos de generación de claves OTP en algoritmos criptográficos considerados obsoletos o “rotos”; un ejemplo de estos son los hashes MD5.

Tercera solución

Como tercera solución se ha tenido en cuenta la implantación de una infraestructura PKI completa.

Este tipo de servicios son proporcionados por distintas empresas, como por ejemplo Realsec, la cual proporciona el hardware para la generación, comprobación y mantenimiento de certificados X.509.

En este caso se deberían valorar los costes de diversas empresas, las cuales proporcionan tanto la solución completa, yéndose totalmente de presupuesto por proporcionar hardware y mantenimiento, como dispositivos para el almacenaje de certificados (smartcards, Token USB).

Desde el punto de vista de los dispositivos cabe evaluar el soporte y la solución que éstos proporcionan, y sobre todo, si no tienen problemas de cara al agente que gestionará la formación del canal SSH desde el lado del cliente.

A pesar de ser la solución más costosa, se presupone que al ser una instalación completa y mantenida por una empresa experta en ello, todos los puntos a tener en cuenta se verían solucionados. Sin embargo, sería una solución difícil de alcanzar, pues los costes tienden a ser excesivamente caros, desembocando en la necesidad de hacer un estudio de mercado para encontrar la mejor opción.

Cuarta solución

La cuarta solución valorada ha sido una opción que implica cierto desarrollo, aunque probablemente podría ser la opción más rentable frente a las anteriores ya mencionadas.

Esta solución se basa en la implantación, bajo previa adaptación, de la solución proporcionada por Google con su Google Authenticator, cuyo servicio proporciona un sistema two factor authentication sustituyendo el uso de los certificados por el de un sistema OTP.

Con esta cuarta solución se podría disponer de un sistema two factor authentication sin necesidad de realizar una fuerte inversión.

El problema de este sistema viene de que, si bien tiene soporte para los servicios de Google según ellos afirman, presenta todos y cada uno de los problemas mencionados en los puntos anteriores; sustituyendo el de los costes, ya que el servicio sería gratuito, y el dispositivo a emplear sería cualquier dispositivo móvil.

Al igual que en las dos primeras soluciones, se debería encontrar un agente SSH capaz de proporcionar el acceso mediante OTP y two factor authentication, en todas y cada una de las aplicaciones y servicios que se quieran emplear en CeSViMa.

Sin embargo, el lado positivo de esta solución sería observándola de cara a implantar una solución propia para el CeSViMa. Si se pudiese encontrar una solución al problema de los agentes SSH, o si en el peor de los casos se debiera entrar a la adaptación de alguno, se podría optar a esta solución, pues Google proporciona el código de su sistema, lo que podría facilitar el diseño de la solución.

Quinta solución

Como quinta y última solución, es decir, en caso de que ninguna de las soluciones se vea factible de implementar, nos encontraríamos en el peor de los casos.

El proyecto se encontraría en el punto en que, ningún agente se puede considerar adecuado de cara al soporte de los requisitos pedidos, además de que la modificación o rediseño de alguno se considere inviable; que los costes proporcionados por las distintas empresas para sus productos sean excesivamente elevados; y que todos los dispositivos diseñados para sistemas PKI no se comporten o den solución a todos y cada uno de los puntos pedidos.

En este último caso se buscaría diseñar, sin llegar a la implementación por salirse de la envergadura del proyecto, de una infraestructura PKI basada en soluciones libres, como pueden ser librerías OpenSSH, o de bajo coste. Además se deberá evaluar y diseñar una posible solución al problema del agente SSH, bien buscando el desarrollo de uno a medida, o bien realizando una interfaz capaz de mitigar y dar solución a los aspectos pedidos por el CeSViMa, la cual deberá dar soporte a los múltiples sistemas operativos que los usuarios del CeSViMa emplean.

6. Adaptación e implementación de soluciones

En este apartado **se tratará de definir la solución más óptima y aproximada a las exigencias del CeSViMa; ya que por falta de tiempo y recursos no se ha podido llegar a una fase de implementación de la solución**, es decir, el trabajo se ha quedado en el estudio de los distintos sistemas de autenticación alternativos a passwords, y además la definición y explicación de la que sería la mejor solución para el problema planteado.

Dado que el proyecto se ha quedado en un estudio y búsqueda de la mejor solución, se ha tratado de encontrar los sistemas más económicos, obviando tanto las implementaciones PKI completas que ofrecen empresas a un coste muy elevado, como las soluciones biométricas. Las soluciones biométricas representan la mejor opción de cara a la autenticación de un usuario, pues son características únicas del mismo; sin embargo, ya se trató en el apartado *2.6. Sistemas biométricos para acreditación de acceso a usuarios* cómo, los sistemas biométricos considerados como "seguros" son aquellos basados en escáneres oculares, que actualmente son extremadamente costosos.

Por este motivo, la búsqueda de la solución se basó en las propuestas primera y cuarta, planteadas en el apartado *5. Diseño de solución*.

- En la **primera solución** se planteó el uso de Smart Cards o Token USB, como almacenes de certificados y claves privadas en una infraestructura PKI. El principal problema que ésta solución tenía era la de ser un sistema multiplataforma.
  En términos más técnicos y concretos el problema que planteaba era que, mediante sistemas o librerías gratuitos y multiplataforma, como por ejemplo las librerías OpenSSH y la suite Putty para Windows, no había ningún medio para poder emplear Smart Cards o Token USB como almacenes de certificados; existía un bug [19] donde se declaraba que, con estas librerías, no se permitía la introducción de PIN para el desbloqueo de Smart Cards y Token USB. Esto provocaba que tras realizar un mandato ssh o scp pertenecientes a la librería OpenSSH, la petición de contraseña no se correspondía con el PIN, pues la librería no poseía soporte para la obtención de certificados de estos dispositivos.

- En la **cuarta solución** se planteó el uso de un sistema One Time Password; pero estos sistemas plantean el mismo problema que la primera solución, y es que es difícil encontrar un sistema o solución multiplataforma. Aunque también cabe mencionar que este tipo de soluciones está empezando a ser bastante empleado en muchos sistemas de autenticación, pudiendo ser más sencillo encontrar una solución multiplataforma que posea interfaz de desarrollo (SDK), lo que permitiría desarrollar una solución adaptada al CeSViMa.

En los siguientes sub apartados se tratarán de adaptar estas dos soluciones al problema focalizado en el CeSViMa, pues son las más viables dentro del marco de soluciones estudiado a lo largo de este documento.

6.1. Adaptación de la primera solución

En la primera solución se tiene el problema de la adaptación de la librería OpenSSH, la cual se emplea para la construcción de lo que se conoce como Secure Shell, es decir, un canal SSH entre un cliente y un servidor en base a una terminal o Shell; además es la librería empleada por muchos otros clientes gráficos para crear canales seguros de comunicación.

Esta librería tiene la ventaja de ser de libre distribución, es decir, es gratuita. Por este motivo se considera el camino correcto del que partir para generar la infraestructura PKI con Smart Cards o Token USB, ya que son librerías muy testeadas con menor posibilidad de fallos.

El problema que plantea esta librería viene a la hora de realizar comunicaciones con dispositivos como Smart Cards y Token USB, ya que estos dispositivos son almacenes seguros de certificados y clave privadas, por lo que requieren de un PIN o clave de desbloqueo. OpenSSH dispone de un sistema de petición de claves, pero no es capaz de realizar la comprobación posterior del certificado o clave privada del usuario.

La peculiaridad que poseen estos dispositivos por la petición de PIN, conlleva a que el uso de mandatos como "*ssh -I /usr/lib/opensc-pkcs11.so*", contenidos en OpenSSH para el acceso a dispositivos sin PIN, no funcionen correctamente. Del mismo modo, el modificar el fichero de configuración de ssh contenido en "*/etc/ssh/ssh_config*" o "*${HOME}/.ssh/config*", incluyendo la línea "*PKCS11Provider /usr/lib/opensc-pkcs11.so"* para que realice la petición de certificado, tampoco funciona. OpenSSH intenta buscar claves en los dispositivos, pero al no "saber" pedir PIN deduce que no hay certificados y falla. Por ello, hasta el momento se tendía a emplear el siguiente procedimiento empleando **ssh-agents** como intermediario para el uso de certificados contenidos en Smart Cards y Token USB.

- En primer lugar se extrae la clave pública del certificado en cuestión. Para ello se obtiene el ID del certificado del dispositivo.

  ```
  $ pkcs15-tool -c
  ```

- En segundo lugar se extrae la clave pública del certificado a raíz de su ID, almacenándola en un archivo temporal. Se pedirá el PIN correspondiente al desbloqueo del dispositivo.

  ```
  $ pkcs15-tool --verify-pin --read-ssh-key ID_CERT -o /tmp/auth_pub.key
  ```

- En tercer lugar se exportará la clave pública, almacenada en el archivo temporal, en el host remoto con el que se pretende establecer una conexión ssh.

  ```
  bash$ scp /tmp/aut_pub.key user@host:/home/user/.ssh/authorized_keys
  ```

- Posteriormente, cada vez que se desee establecer una conexión ssh con dicho servidor, bastará con realizar los siguientes pasos:

    - Se arrancará un agente ssh, cargando posteriormente la biblioteca PKCS correspondiente, por ejemplo pkcs#11.

      ```
      $ ssh-agent /bin/sh
      sh$ ssh-add -s /usr/lib/opensc-pkcs11.so ( introducir PIN )
      ```

    - Se accederá mediante el cliente que se desee al servidor, en función del servicio que se pretenda realizar (ssh, slogin, scp, sftp), por ejemplo con ssh sería:

      ```
      sh$ ssh user@host
      ```

- Por último tan solo se deberá cerrar sesión, terminando así la conexión ssh.

Sin embargo, y a pesar de no estar reconocidos por el equipo de desarrollo de OpenSSH, existen diversos parches para solventar el problema ya comentado. Un buen ejemplo de este tipo de parches es el conocido como OpenSC, el cual está también incluido en una suite específica para Putty, el cliente más extendido en el entorno Windows para el establecimiento de canales SSH entre cliente y servidor.

Para las plataformas Linux y Mac OS X, se debe recompilar la librería OpenSSH con la opción "*configure --with-opensc=/PATH_OPENSC_LIB*", después se deberá añadir el parche incluido en el directorio "*opensc-0.9.6/src/openssh/ask-for-pin.diff*". De esta forma bastará con ejecutar el mandato "*$ ssh -I 0 user@host*" para especificarle al cliente ssh que deberá obtener el certificado del lector con id 0 (es el id por defecto que poseen al conectarse en un ordenador); de esta forma se establecerá una conexión SSH mediante el certificado almacenado en una Smart Card o Token USB, pidiendo previamente el PIN de acceso al dispositivo. Para los usuarios que empleen ssh-agents, tiene el añadido de poder incluir el PIN en el agente mediante el mandato "*ssh-add -s 0*"; de esta forma no será necesario introducir el PIN en cada conexión.

Para las plataformas Windows se encuentra la extensión **PuttyCard** [20], la cual contiene el parche OpenSC, siendo posible el uso de dispositivos almacenes de claves como las Smart Cards y Token USB.

Con estos procedimientos es posible la adaptación de la primera solución propuesta, siendo por tanto viable la implementación de una infraestructura PKI empleando recursos gratuitos, ya que las Smart Cards que se empleen pueden ser los DNIe que todo ciudadano español posee por obligación.

6.2. Adaptación de la cuarta solución

La cuarta solución propuesta consistía en la implementación de un sistema One Time Password (OTP); una explicación más extendida se encuentra en el apartado *4. Informe de vigilancia sobre soluciones propuestas en el mercado actual*. En este apartado se explicó el funcionamiento de estos sistemas, así como la necesidad de una **sincronización entre cliente y servidor para el mutuo conocimiento de las claves de acceso aleatorias y temporales**.

El problema de estos sistemas viene a ser similar al de la primera solución; y es que las empresas que promocionan estos servicios, como es RSA, proporcionan no solo los dispositivos USB con display para mostrar la clave al cliente, sino también el hardware necesario para la sincronización en el lado del servidor. Son servicios que también se exceden en presupuesto, y como ya se ha comentado al inicio del apartado, se trata de buscar soluciones de coste reducido y/o gratuito.

Por este motivo se ha hecho hincapié en la solución propuesta por Google, con su Google Authenticator. Pero además se mostrará en este sub apartado la solución, aún más interesante desde el punto de vista del autor, propuesta por la empresa española Eleven Paths (subcontrata de Telefónica encargada de la seguridad de la información) con su proyecto conocido como Latch.

En **primer lugar Google**, con su Google Authenticator [21], ha mostrado con todos sus servicios, tales como Gmail y Youtube, cómo **su proyecto permite la inclusión de un sistema de autenticación One Time Password al clásico sistema de usuario y contraseña** que ya poseían los mismos.

Google tiene disponible parte del código de su proyecto [22], exceptuando el código referente a sus aplicaciones. Con este código, Google pretende que los desarrolladores diseñen sistemas similares al suyo, mejorando así la autenticación de cualquier otro servicio.

La ventaja de este proyecto es que el dispositivo que requiere es un smarphone, tanto Android, como iOS y Blackberry, para la obtención de las passwords aleatorias y temporales. Además la parte del servidor no requiere de ningún tipo de hardware, sino la configuración y desarrollo del código proporcionado por Google; de esta forma se tiene una estructura OTP de forma gratuita y sin necesidad de portar con otro dispositivo más.

Por otro lado está el proyecto desarrollado por Eleven Paths (empresa española), quienes con su proyecto **Latch** están tratando de introducir sistemas de autenticación más seguros.

El proyecto conocido como Latch [23] propone la implementación de un “pestillo” (en Inglés Latch) a modo de bloqueo de acceso a un servicio, es decir, **propone el bloqueo de identidades digitales cada vez que no se quiera hacer uso de ellas**; todo desde el dispositivo móvil. Otorga además **la posibilidad de recepción de avisos al dispositivo móvil asociado, cada vez que se intente acceder a una cuenta bloqueada**, sirviendo de advertencia al usuario de que alguien que conoce sus credenciales correctos, quiere acceder a su cuenta. Latch **implementa la opción de incluir un sistema de seguridad One Time Password** con las cuentas del usuario.

Con todo esto, se tiene que Eleven Paths proporciona un servicio de gestión de cuentas bastante completo.

El funcionamiento de Latch es simple. Tras registrarse en su servicio web y descargarse su aplicación para móviles, tan solo se requiere acceder a un servicio con soporte para Latch y vincular el dispositivo móvil. Según los propios desarrolladores, ni el usuario ni la contraseña de Latch son enviados en ninguna ocasión al servidor, tal y como se puede ver en la *imagen 14*, donde se representa su funcionamiento.

Pero no solo servicios con soporte para Latch pueden ser protegidos, pues Eleven Paths proporciona una API REST para configurar cualquier servicio web, así como SDKs para los lenguajes de programación más populares, como PHP, Python, Java, .NET, Ruby, y C. Además poseen plugins para los gestores de contenidos más usados, como Joomla, Wordpress, y Drupal.

Para la protección de un servicio con Latch, es necesario el registro de la aplicación (*imagen 15*), creando un Application ID y un Secret Key, que son necesarios para poder realizar llamadas a la propia API.

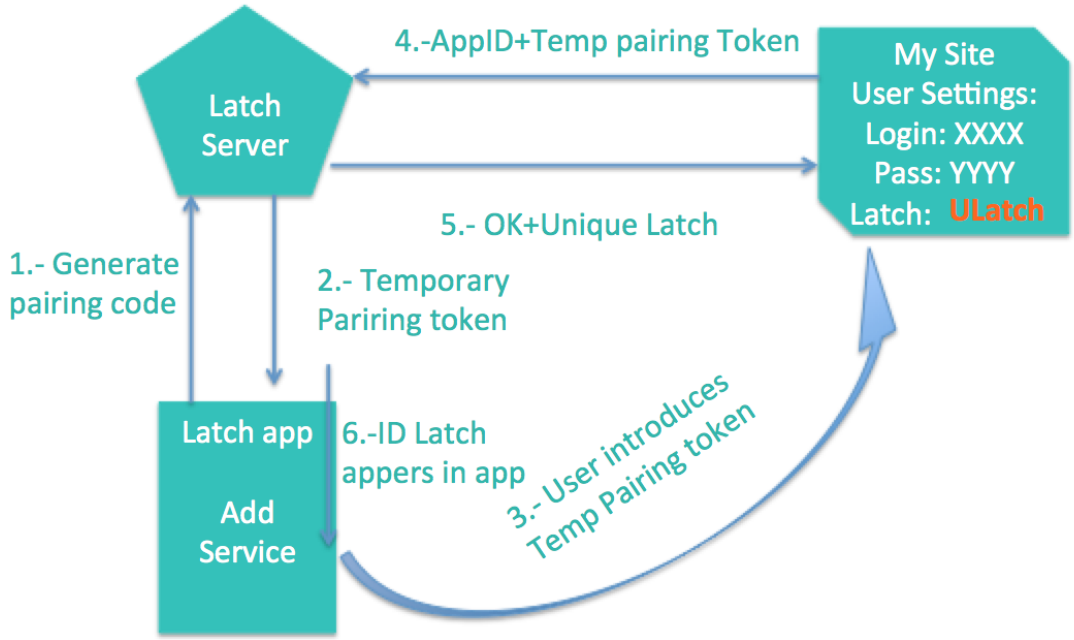

Imagen 14, proceso de establecimiento de una identidad en Latch

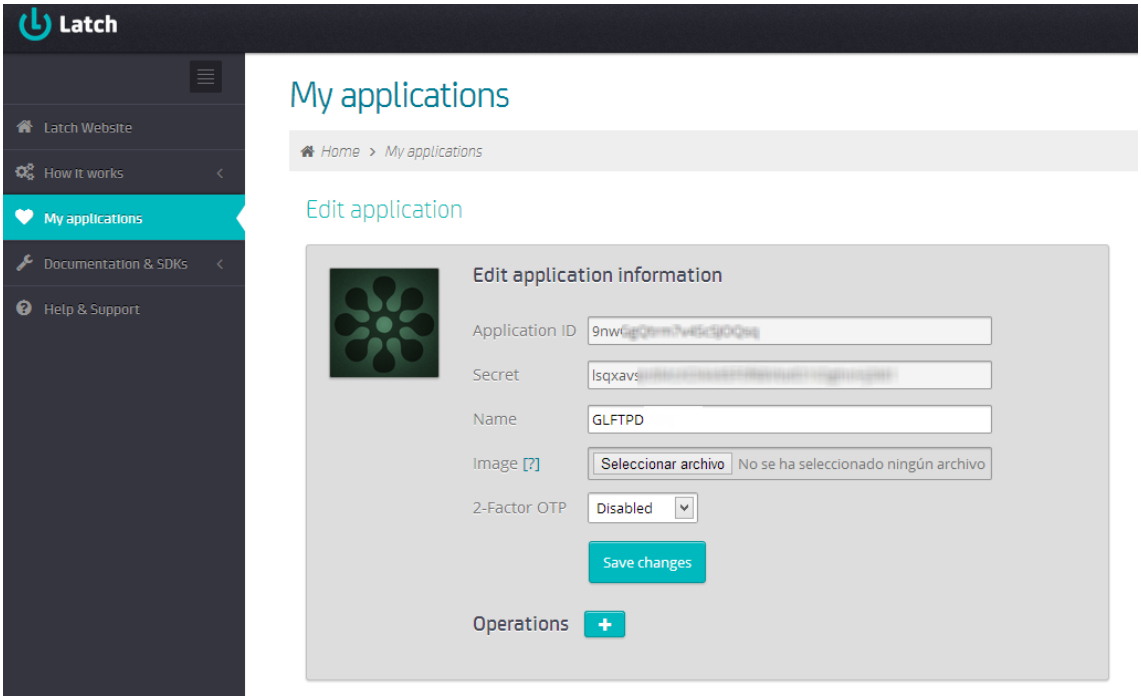

Imagen 15, pantalla de administración de aplicaciones

El proceso de emparejado en Latch consiste en la petición de emparejamiento previa en la parte del servidor, la cual generará un token temporal para introducir en el dispositivo móvil. En la *imagen 16* se puede ver el proceso de emparejamiento en el lado del cliente.

Por último, como se puede ver en la *imagen 17*, al usuario se le muestra en el dispositivo un listado con los servicios que tiene emparejados, donde con un solo clic podrá bloquear o desbloquear el acceso al servicio, como si de un pestillo se tratase.

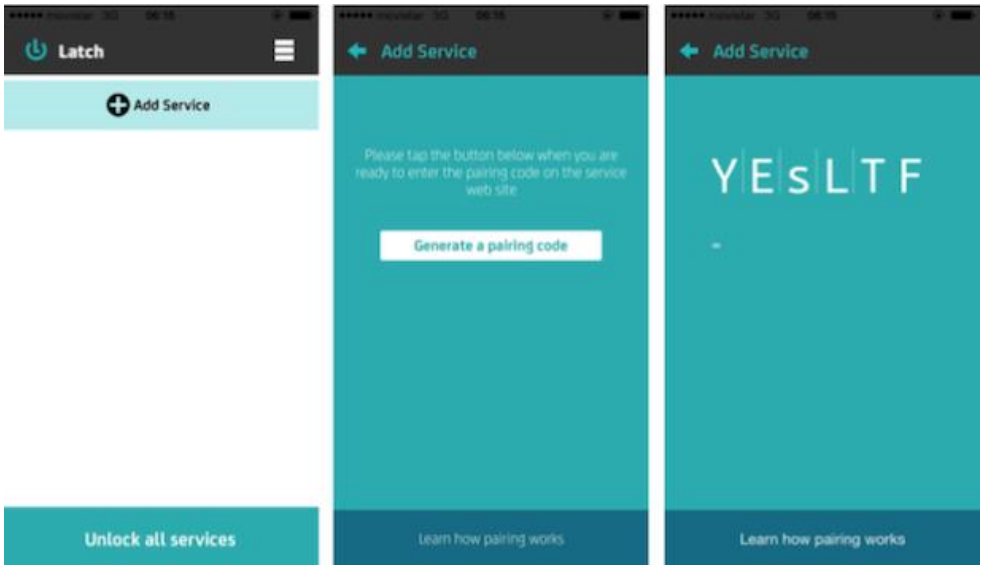

Imagen 16, proceso de emparejamiento desde el cliente

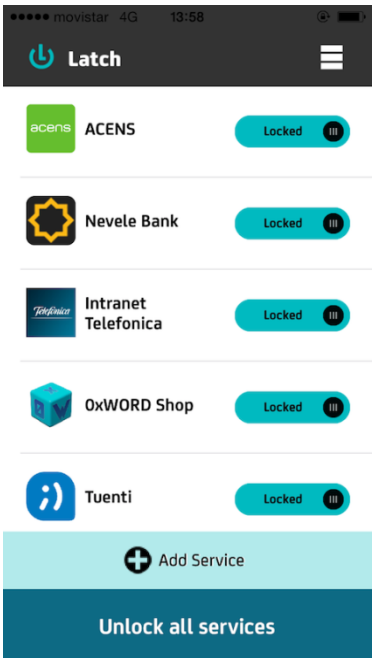

Imagen 17, listado de aplicaciones emparejadas

En caso de tener activo el uso de One Time Password, en la misma aplicación se proporciona la clave de acceso o token temporal, como se puede ver en el ejemplo de la *imagen 18*.

Imagen 18, ejemplo de petición de OTP con Latch

Todas las imágenes mostradas relacionadas con el proyecto Latch han sido recogidas de la web del propio proyecto, incluida en la bibliografía, así como del blog de uno de los directores de desarrollo del proyecto, Jose María Alonso (www.elladodelmal.com).

6.3. Conclusión de las adaptaciones

Tras haberse definido la adaptación de esas dos soluciones, tres si se diferencian entre las dos planteadas dentro de la cuarta solución, puede verse cómo es posible la implementación de las mismas sin llegar a contratar una solución a un alto coste.

En cuanto a la mejor opción, entre la primera y cuarta solución, depende de diversos factores:

- **Privacidad:** La primera solución otorgaría una mayor privacidad a la solución llevada a cabo en el CeSViMa; ya que las adaptaciones referentes a la cuarta solución requieren de una autenticación previa de la aplicación en los sistemas de Google o Eleven Paths.

- **Sencillez de implantación:** La sencillez desde el punto de vista de la implantación de las tres adaptaciones es relativa al desarrollador. Al no haberse podido probar la adaptación de la primera solución, no es posible saber si sería necesaria una modificación en la configuración del parche; de no ser necesaria, la primera solución sería la más sencilla de implantar en el centro. En caso contrario dependerá de las preferencias del personal del CeSViMa, y la habilidad del desarrollador; aunque habría que evaluar también los cambios a realizar y añadir en caso Google Authenticator, pues no garantizan una sencilla implantación.

- **Seguridad y usabilidad:** Al evaluar la seguridad de un sistema de autenticación, se tiende a pensar que se pierde en términos de usabilidad. En este caso se han planteado tres soluciones bastante distintas.

    - La primera solución proporciona un sistema Two Factor Authentication con empleo de Smart Cards y/o Token USB; lo que proporcionaría un sistema fiable, seguro, y donde el usuario tan solo tendría que insertar su Smart Card en el lector (siempre y cuando la adaptación funcione correctamente). Se puede considerar un sistema de autenticación con un alto nivel de usabilidad, pues la Smart Card puede estar conectada durante toda la sesión de trabajo.

    - La segunda adaptación se refiere al sistema One Time Password con empleo del SDK de Google. Este sistema proporciona una infraestructura Two Factor Authentication mediante el uso del dispositivo móvil, que es donde el usuario obtiene la clave temporal. Puntos en contra de este sistema son, claramente, la usabilidad de cara al usuario, ya que deberá comprobar su dispositivo móvil en cada acceso; además de la posible dificultad en la denegación de servicio por parte del centro, ya que se desconoce la configuración requerida para denegar el acceso a un usuario que el centro desee, por ejemplo, despedir.

- La tercera adaptación corresponde al sistema One Time Password mediante el uso del proyecto Latch. Este sistema, al igual que el anterior, propone el uso del dispositivo móvil para la autenticación del usuario, lo cual es un punto negativo de cara a su usabilidad. Sin embargo tiene un punto a favor de cara a las otras dos soluciones, y es que permite que el propio usuario bloquee el acceso a su cuenta, evitando así que de ninguna forma un tercero pueda acceder a ella. Por otro lado, se conoce que el servidor guarda constancia de los usuarios que tiene registrados, pudiendo bloquearse desde el mismo el acceso a cualquiera de las cuentas, lo que sería de ayuda en el ejemplo anterior del despido de un usuario.

En estos apartados se ha tratado de señalar los principales puntos, a tener en cuenta, de cara a la elección de un sistema de autenticación de entre las soluciones propuestas. La elección final depende del centro CeSViMa, escapándose de la opinión del autor de este documento, la cual se ha tratado de resumir y expresar de la mejor forma posible en este sub apartado.

7. Comparación y análisis de resultados frente a métodos tradicionales

Con el apartado *2.2. Autenticación mediante passwords y problemas en su uso*, se ha tratado de transmitir los problemas que trae el uso de sistemas de autenticación basado en usuario – password. Se ha tratado, además, de definir posibles soluciones y alternativas a las clásicas passwords en el apartado *5. Diseño de solución*; mientras que en el apartado *6. Adaptación e implementación de soluciones* se ha intentado explicar, de la mejor forma posible, la adaptación de las dos soluciones más interesantes para el centro CeSViMa, detalladas en el apartado 5 de este documento.

Con todas las soluciones redactadas en el apartado 5, es posible evitar en gran medida los problemas que traen las clásicas passwords, aunque en este apartado se hará referencia a las adaptaciones explicadas en el apartado 6, es decir, a las dos soluciones más factibles de entre las propuestas en el apartado 5.

Los sistemas de autenticación basados en usuario – password son sistemas muy sencillos de implementar, y aparentemente usables para los usuarios. Pero no es del todo cierto, ya que de intentar mejorar la seguridad de estos sistemas los problemas de usabilidad crecen de forma exponencial para los usuarios. Éstos deben ser capaces de recordar cientos de passwords largas y complejas, y a ser posible sin tener ninguna relación de entre las establecidas en el resto de entidades digitales.

Con las soluciones propuestas, tanto con el sistema PKI con Smart Card, como con las basadas en sistemas One Time Password, **el usuario (en este caso exclusivo del CeSViMa) no debe memorizar ninguna password**; como mucho un PIN numérico para el bloqueo de la Smart Card o Token USB, el cual en caso de pérdida o intervención por parte de una tercera persona no sería una catástrofe para la empresa. En caso de que un tercero conozca el PIN de desbloqueo de estos dispositivos, requerirá además el propio dispositivo; en caso de extravío o robo del mismo la empresa puede denegar su acceso de forma inmediata.

La seguridad mediante el uso de estos sistemas se ve aumentada considerablemente. Se fortalece el eslabón más débil de la cadena de seguridad en la autenticación, el usuario, el cual es vulnerable a ataques de ingeniería social, o puede olvidar la password. **El servidor es en quien recae la mayor parte de la seguridad**, y donde se centrarán todos los ataques; en el caso de emplear una infraestructura PKI, intentando obtener su clave privada y/o certificado de confianza; en el caso de un sistema OTP, averiguando el algoritmo de generación de claves de acceso.

Concluyendo este apartado, con el uso de estos sistemas de autenticación **se gana en usabilidad; pues los usuarios no deben recordar complejas passwords, y tan solo requieren de un dispositivo para que realice el proceso de autenticación de forma segura**. Pero **además se gana en seguridad**, pues recae una mayor parte de la seguridad del proceso en el propio servidor, en la entidad o empresa, quitándosela a los usuarios que, en la mayor parte de las ocasiones, no están correctamente formados o no tienen por qué tener dicha carga.

8. Conclusiones y líneas futuras

En este trabajo se han estudiado distintas vías para la resolución del problema planteado por el centro CeSViMa. Estas soluciones han sido comentadas y redactadas en el apartado *5. Diseño de solución*, para centrarse posteriormente en la adaptación de las dos más adecuadas en el apartado *6. Adaptación e implementación de soluciones*.

Durante el desarrollo de este documento, se ha tratado de recoger y expresar de la mejor forma posible las distintas soluciones, e implementaciones, llevadas a cabo actualmente para la sustitución de los sistemas de autenticación basados en usuario - password. Pero además se ha podido observar, no solo los muchos problemas que acarrea el emplear los sistemas basados en passwords, sino que no se puede considerar ningún sistema de autenticación como realmente seguro; pues siempre es posible encontrar o provocar una falla de seguridad en dichos sistemas, tal y como se puede ver en el *apéndice II* con algunos de los muchos tipos de ataques.

Mirando de cara a líneas futuras de este proyecto, **lo primero que habría que conseguir es la implantación de una de las dos soluciones que se han podido adaptar**; pero **posterior a esto, se debería realizar una fuerte evaluación de la misma, sobre todo de cara a los usuarios** que deban utilizar el nuevo sistema de autenticación, pues son quienes deben sentirse cómodos y seguros empleándolo.

Aun así, no sería una auténtica finalización del trabajo, pues siempre se pueden mejorar todos los sistemas. Son entornos que cambian constantemente, y tal y como se ha podido leer en este documento, cada día surgen nuevas técnicas y medios de ataque para todos los sistemas.

A modo de conclusión de este apartado, **el establecer un sistema de autenticación considerado seguro, así como poseer un sistema que se considere seguro, no es una tarea que finalice nunca. Pues se debe actualizar e ir modificando constantemente, ya que es una carrera frente a los cibercriminales que posiblemente nunca acabará**.

De cara a un proyecto de continuación, sería recomendable la implantación de un sistema que sea fácilmente actualizable y extensible. Se debe tener en cuenta que cualquier cambio en un servicio, en este caso un sistema de autenticación, debe ser lo más suavizado posible para no causar molestias a los usuarios del mismo. En este caso se tratará de un cambio radical en el sistema de autenticación, pero en caso de escoger la implantación de un sistema PKI, posibles actualizaciones podrían consistir en el cambio del tipo de certificados, lo que no perjudicaría a los usuarios. Por otro lado, el reforzar un sistema OTP, consistiría en el cambio del algoritmo de generación de claves, o incluir un hardware para generar entropía y aleatoriedad a la hora de generar claves en el servidor, lo que tampoco percibirían los usuarios. Con esto se podría disponer de un sistema que, sin variar la forma o vías de autenticación, se podría ir reforzando en el día a día, consiguiendo un sistema lo suficientemente seguro de cara a las futuras amenazas.

9. Apéndice I: Conceptos básicos de criptografía

9.1. Comunicaciones seguras: Protocolos SSL y TLS

A menudo se habla de canales seguros en las comunicaciones entre dos puntos, haciéndose mención a los protocolos ***Secure Socket Layer*** (SSL) y ***Transport Layer Security*** (TLS).

En este apéndice se tratarán de explicar los principales aspectos de estos protocolos, tanto su funcionamiento como su evolución, pues se ha considerado de interés para el autocompletado de la memoria de este proyecto.

Ambos protocolos son protocolos criptográficos desarrollados para el establecimiento de canales o comunicaciones seguras en redes, siendo la principal de las redes Internet. De hecho TLS es la evolución de SSL, siendo también conocido como SSL 3.1.

El protocolo SSL proporciona autenticación y privacidad en la información enviada entre el emisor y receptor de una comunicación. Generalmente sólo el servidor es autenticado, garantizando su identidad, y manteniéndose el cliente sin autenticar. El protocolo SSL consta de cuatro fases básicas:

- Negociado del algoritmo criptográfico a emplear en la comunicación por las dos partes
- Intercambio de las claves públicas y certificados digitales, proporcionando así autenticación.
- Intercambio de las claves para el algoritmo criptográfico simétrico elegido por las dos partes. Este intercambio se encuentra protegido gracias al cifrado asimétrico mediante las claves púbicas ya intercambiadas.
- Cifrado del resto de la comunicación basado en el cifrado simétrico/asimétrico seleccionado con las claves intercambiadas.

Las implementaciones actuales de SSL permiten el uso de distintos algoritmos criptográficos a emplear para cifrar el tráfico de la comunicación:

- En el caso de emplear criptografía asimétrica, se encuentran algoritmos como RSA, Diffie-Hellman, DSA (Digital Signature Algorithm), o Fortezza.
- Si se emplean algoritmos basados en criptografía simétrica, se encuentran algoritmos como RC4, IDEA (International Data Encryption Algorithm), DES (Data Encryption Algorithm), Triple DES y AES (Advance Encryption Algorithm).

El protocolo SSL, desarrollado originalmente por Netscape, posee una versión 1.0 que nunca fue entregada públicamente. Sí fue publicada la versión 2.0 en febrero de 1995, pero debido a la gran cantidad de fallas de seguridad que contenía fue totalmente rediseñada en su versión 3.0, publicada en 1996.

Por otro lado, como ya se ha mencionado en este apartado, el protocolo TLS 1.0 fue una actualización del SSL 3.0. TLS posee otras dos versiones posteriores, siendo estas la 1.1, a la que se añadió protección contra los ataques de algoritmos de cifrado por bloques (CBC), y la versión 1.2, que mejora aspectos de compatibilidad con SSL, así como el uso de algunos de los algoritmos de cifrado que soporta.

Funcionamiento

El protocolo SSL intercambia distintos registros, pudiendo ser cada uno de ellos comprimido, cifrado y empaquetado previamente con la MAC (Message Authentication Code).

Al inicio de cada conexión SSL se produce el protocolo *handshake* (protocolo de acuerdo), constando de las siguientes fases:

- Envío del *ClientHello*, donde se especifica la lista de algoritmos de cifrado, compresión, y versión más alta del protocolo SSL permitido. Se envían además bytes aleatorios, llamados *reto*, que se emplearán más adelante.

- El cliente recibe un mensaje *ServerHello*, donde el servidor ha escogido entre los parámetros enviados previamente.

- Cuando todo ha sido recibido, se procede al intercambio de los certificados, siendo actualmente certificados X.509.

- Por último, cliente y servidor negocian una clave secreta llamada *master secret*. Este intercambio es realizado mediante el uso de algoritmos como Diffie-Hellman, o mediante el cifrado de clave pública y privada de cliente y servidor.

En la *imagen 19* se puede ver la representación gráfica del protocolo *handshake* que se acaba de detallar.

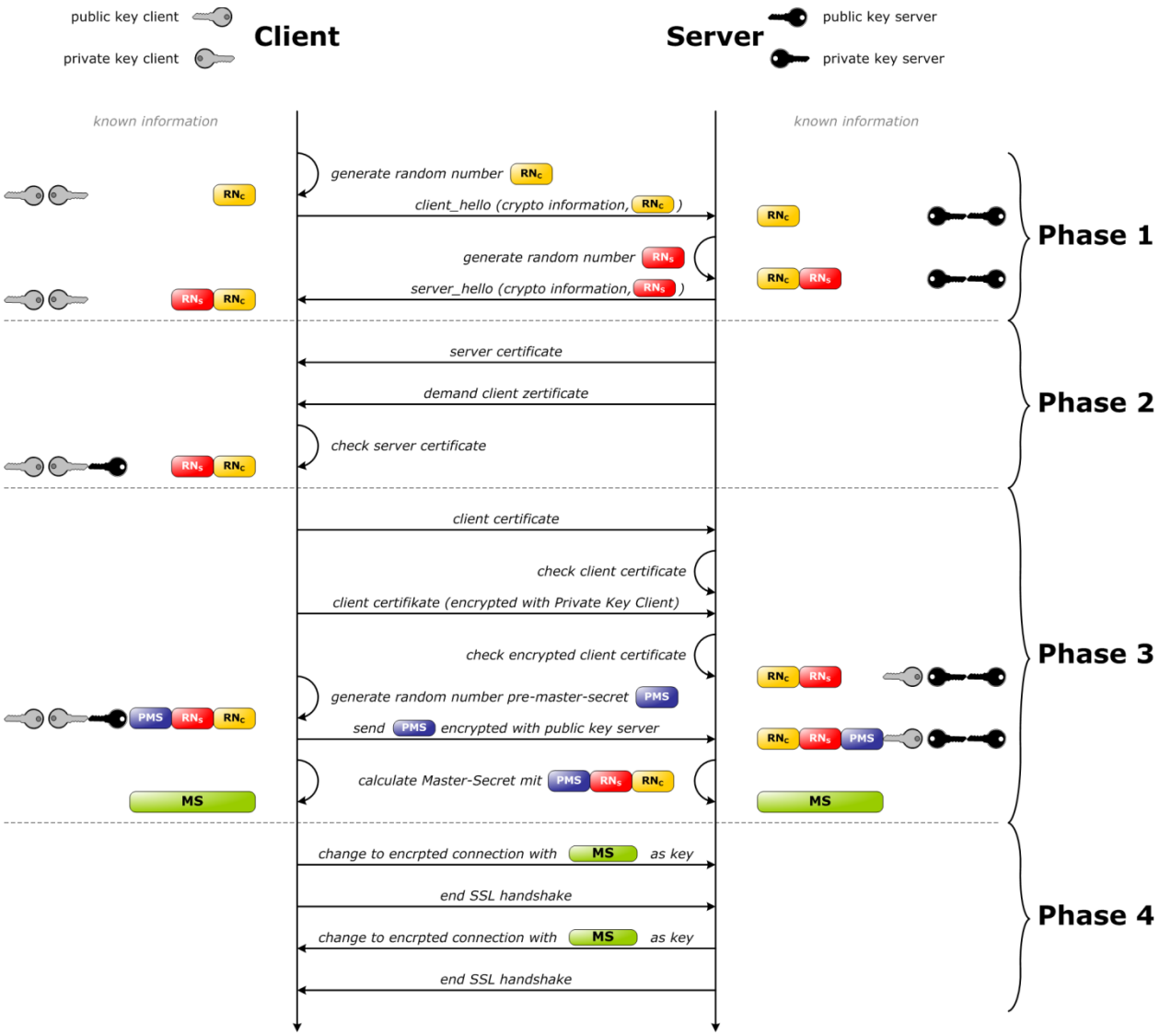

Imagen 19, Representación gráfica del protocolo *Handshake* obtenida de Wikimedia

Aplicaciones

Estos protocolos se ejecutan en una capa entre los protocolos de aplicación HTTP, SMTP, NNTP, y sobre el protocolo de transporte TCP. Proporciona una capa de seguridad en estos protocolos, dando lugar al renombrado de alguno de los mismos, como puede ser HTTPS.

HTTPS es el protocolo empleado para asegurar las páginas de la World Wide Web, sobre todo en las páginas referentes a comercio electrónico. Tienden a emplearse certificados de clave pública como los vistos en este documento, obteniendo así la autenticación y no repudio en el proceso de compra online.

SSL puede ser también empleado para tunelizar una red completa, creándose así una red privada virtual (VPN), esto es empleado en librerías como pueden ser OpenVPN.

El principal problema de su uso viene en el coste computacional que a menudo requiere, pues para determinados usos o servicios puede ser muy costoso el establecimiento de un canal seguro mediante TLS/SSL y su protocolo *handshake*. Sin embargo, día a día todos los navegadores y servicios tienden a adoptar su uso, pues mejoran en gran medida la seguridad en los procesos de autenticación.

10. Apéndice II: Posibles ataques a sistemas protegidos por passwords

En este apéndice se tratará de explicar de forma superficial, y a modo de concepto, algunos de los muchos ataques empleados para la obtención o garantización de acceso de terceras personas a un sistema o servicio.

La mayoría de ellos son empleados en todo tipo de sistemas, entre los que se incluyen los que incorporan autenticación mediante passwords. Sin embargo, tal y como se ha comentado a lo largo de este documento, hay distintas soluciones para dificultar algunos de estos ataques. Todos estos detalles, incluyendo la posible contra medida para el caso planteado en el CeSViMa, serán explicados en cada uno de los siguientes sub apartados.

10.1. Brute force attack

Los ataques de fuerza bruta fueron los primeros ataques empleados en el criptoanálisis, consistiendo este tipo de ataques en la **evaluación de todas las combinaciones posibles hasta encontrar la que dé acceso al sistema**.

Inicialmente se basan en el conocimiento del algoritmo de cifrado empleado y una serie de pares de textos en claro y cifrados, correspondientes entre sí; a mayor número de pares de textos cifrados y descifrados más eficiente es el ataque.

El esfuerzo requerido para que el ataque sea exitoso equivale a $2^n - 1$ operaciones, siendo ***n*** la longitud de la clave o *espacio de claves*. Con esto se tiene que un factor determinante en el coste de este tipo de ataques, radica en la longitud de la clave, incluyéndose en ellas caracteres numéricos, alfanuméricos y símbolos para lograr aumentar dicho espacio de claves.

Este tipo de ataques se suele combinar con los llamados ***ataques de diccionario***, que consisten en la previa comprobación de palabras o claves ya registradas previamente; es decir, con los ataques de diccionario se eliminan previamente las contraseñas más comunes empleadas, reduciendo significativamente la complejidad de un ataque de fuerza bruta.

Los ataques de fuerza bruta se ven mejorados por el fuerte incremento de la potencia de cálculo que tienen las máquinas actuales. Tiende a emplearse el uso de GPUs para su realización, así como varias máquinas y/o servidores, pudiéndose realizar miles de comprobaciones por segundo.

Para mitigar este tipo de ataques, es recomendable emplear sistemas de autenticación como los vistos en este documento, un buen ejemplo es el uso de los **certificados con claves públicas**. Sin embargo, siempre se suelen erradicar **imponiendo un límite de intentos de sesión**, aunque esto es solo viable cuando se trata de asegurar un sistema de autenticación.

10.2. Phishing attack

El Phishing o suplantación de identidad, es un tipo de ataque caracterizado por emplear lo que se denomina como *ingeniería social*. La ingeniería social consiste en el estudio, y posterior engaño, de los usuarios víctima; se pretende obtener con esto los credenciales de acceso o información confidencial de las víctimas.

El atacante, conocido también como *phisher*, se suele hacer pasar por una persona o entidad de confianza de cara a la víctima. El método de contacto o engaño con la víctima suele ser indirecto y nunca en persona, es decir, mediante correos electrónicos, llamadas telefónicas, o falsos sistemas de autenticación.

Normalmente, los ataques de este tipo suelen tomar como objetivo la obtención de datos bancarios y autenticaciones online en distintos servicios. Por este motivo se suelen emplear emails falsos muy convincentes, suplantando las entidades auténticas en cuestión; de esta forma los clientes o víctimas acceden a entregar los datos pedidos de forma inconsciente a terceras personas.

En otras ocasiones también se suelen falsificar los propios sitios web, copiando en la mayor medida de lo posible el entorno real de acceso. Las víctimas suelen ser redirigidas mediante enlaces falsos, pero muy similares, a estos entornos, de forma que de nuevo ceden sus credenciales de acceso a los atacantes. Algunas técnicas para obtener un enlace de redireccionamiento convincente se basan en el uso del carácter “@”; esta técnica consiste en emitir un enlace de la forma *www.google.com@mi-sitio.com*, donde el enlace dirige realmente a la página *mi-sitio.com*, que habrá sido previamente rediseñado para que aparente ser la del sitio a suplantar.

A pesar de que éstas técnicas de phishing suelen ser actualmente detectadas por los navegadores y diversos sistemas de protección de navegación online, como antivirus, hay métodos más trabajados para realizar esa suplantación.

Se conoce como ***Cross Site Scripting*** (XSS) a la previa inyección de código malicioso en el sitio web en cuestión, de forma que se alteran las peticiones de las credenciales a los usuarios, siendo éstas redirigidas al atacante. También se puede incluir el ataque ***IDN spoofing***, donde el link enviado a la víctima es similar al original, pero reescrito con caracteres homógrafos (dominio.com se ve similar a dominio.com, aunque en el segundo las letras "o" hayan sido reemplazadas por la correspondiente letra griega ómicron, "o").

Este tipo de ataques es posible evitarlo mediante el uso de certificados legítimos y confiables, siempre basándose en técnicas de autenticación mutua. Además hay empresas que se dedican al estudio de las distintas páginas de phishing, incluyéndolas en listas públicas que emplean aplicaciones y antivirus para prevención de los usuarios.

10.3. Troyanos

Se reconoce como troyano a todo software malicioso que se presenta a un usuario víctima como inofensivo, pero que al ejecutarlo le proporciona al atacante acceso total y remoto al sistema de la víctima. El término troyano proviene del legendario caballo de Troya descrito en la Odisea de Homero.

Este tipo de software no se considera como virus informático, pues no tiene por qué ser de carácter perjudicial para un usuario, ya que **pueden ser simples programas de gestión remota de máquinas**. Para que un programa sea considerado como troyano, éste debe garantizar el total acceso y control de un sistema a un tercero, sin que el usuario legítimo del sistema tenga conciencia de dicho acceso.

Un troyano otorga multitud de funcionalidades a un atacante, ya que posee control total del sistema. Un atacante puede acceder a todos los servicios y sistemas que el usuario víctima tenga almacenados en la máquina infectada, así como acceder a toda la información que posea. Pero además, teniendo en cuenta el fuerte crecimiento de los troyanos en el mundo de los dispositivos móviles, los atacantes pueden obtener información de geolocalización y el entorno en que el usuario se encuentra, gracias a los múltiples sensores que estos dispositivos contienen.

Actualmente, los troyanos suelen ser empleados para la obtención de credenciales y datos privados de víctimas, por lo que **las fuentes más comunes de infección son las redes P2P, webs con contenido ejecutable, aplicaciones de dudosa credibilidad, archivos adjuntos en correos electrónicos, y métodos de ingeniería social**. En general, en cualquier entorno con contenido ilícito y/o ilegal de internet es una probable fuente de infección de troyanos.

Las mejores opciones para evitar ser víctimas de este tipo de software malicioso suelen ser, como casi siempre, poseer un sistema antivirus, sistema operativo y firewall actualizados, así como ser cuidadoso a la hora de utilizar Internet.

De cara a un sistema de autenticación, motivo que se ha ido tratando a lo largo de este documento, **la mejor opción para evitar la captura de credenciales es evitando que éstos lleguen a la máquina, evitando así que puedan ser capturados por un troyano**. Sistemas que poseen estas características son las infraestructuras PKI que se apoyan en sistemas biométricos, o el uso de certificados gestionados por Smart Cards o Token USB, ya que estos medios no permiten la extracción de las claves privadas del usuario (tal y como se puede leer en los apartados 2.7 y 2.8 de este documento).

10.4. Keylogger

Un keylogger es, como su nombre indica, **un registrador de pulsaciones de teclado**. Su funcionalidad es simple, registrar todas y cada una de las pulsaciones de teclado que la víctima realice y almacenarlas en un fichero, o bien enviarlas por internet directamente al atacante.

La finalidad de un keylogger es la de obtener datos sensibles de la víctima, como pueden ser datos bancarios, contraseñas, o cualquier tipo de información privada.

Se pueden encontrar varios tipos de keylogger, aunque se pueden agrupar en dos grandes grupos:

- **Keylogger hardware**: Este grupo congrega los que consisten en adaptadores conectados al teclado, lo que les hace completamente visibles; o dispositivos que se sueldan de forma interna al teclado.

- **Keylogger software**: Este grupo incluye a los que son desarrollados vía software para ser implantados a un usuario víctima por medio de un virus informático o un troyano.
  Contrario a lo que se crea, son de implementación sencilla, pudiendo ser desarrollados mediante lenguajes comunes de programación, lo que suele generar un gran retardo en la escritura del usuario víctima; o los escritos mediante las propias APIs del sistema operativo, haciéndolos más imperceptibles por las víctimas.

Para combatir este tipo de malware existen distintos métodos, donde **se pueden encontrar desde antivirus y software anti-keylogging**, a menudo basados en la consulta de listas con los keyloggers más empleados. **Además de aplicarse sistemas de monitoreo de red, pudiéndose observar si un software desconocido trata de enviar datos a través de la red**.

Tal y como se está comentando, este tipo de malware es diseñado con la clara intención de interceptar claves en sistemas de autenticación, por lo que la primera medida que se suele tomar es la de emplear teclados virtuales para la introducción de las claves. Sin embargo existe también malware para este tipo de teclados virtuales, los cuales son capaces de enviar capturas de pantalla cada vez que el usuario víctima realiza un clic con el ratón.

La forma definitiva de evitar que un keylogger capture una contraseña es, sencillamente, evitar que se deba introducir una contraseña para autenticar a un usuario, tal y como se comenta a lo largo de este documento con los sistemas de autenticación alternativos al uso de passwords.

10.5. SQL Injection

Este tipo de ataques, conocidos como SQL Injection, **se basan en la inyección de código SQL en sistemas que realizan consultas a una base de datos**. Los sistemas vulnerables a este tipo de inyecciones en las consultas a bases de datos, **proporcionan información sensible o privada al atacante**; normalmente se realizan para la obtención de contraseñas, direcciones, teléfonos, y datos privados de los usuarios de un sistema.

Se confirma que se produjo una inyección SQL cuando se ha insertado código SQL no deseado dentro de un código SQL programado, realizando consultas, y modificaciones indebidas en la base de datos.

Dado que este tipo de ataques se realiza con carácter malicioso, es considerado como un problema de seguridad. Es considerado como un fallo de programación, ya que debe ser el propio programador del código SQL original, quien debe emplear los medios otorgados por los distintos lenguajes de programación para evitar este tipo de ataques.

La vulnerabilidad en que se apoya este tipo de ataques radica en la recepción de parámetros del tipo *string* en la consulta SQL original. En el *código 1* se expone un ejemplo de código SQL vulnerable a este tipo de ataques.

```
consulta := "SELECT * FROM usuarios WHERE nombre = '" + nombreUsuario
+ "';"
```

Código 1, ejemplo de código SQL vulnerable

En el *código 1* se ha definido una consulta SQL que, independientemente del lenguaje en que haya sido escrita, está esperando por un *string* en la variable definida como **nombreUsuario**. De esta forma, cuando un usuario realice la consulta de un nombre (por ejemplo Alicia) de la forma esperada por el programa, éste formará una consulta del tipo que se detalla en el *código 2*.

```
consulta := "SELECT * FROM usuarios WHERE nombre = 'Alicia';"
```

Código 2, formación correcta de la consulta

El problema surge cuando un atacante realiza una inyección como la mostrada en el *código 3*, la cual quedará reflejada en el servidor tal y como se muestra en el *código 4*. Con ese ataque, además de realizar la consulta deseada, el atacante eliminará la tabla *usuarios*, y obtendrá el resultado de realizar una consulta a la tabla *datos*, tabla a la cual no debería tener acceso alguno.

```
Alicia'; DROP TABLE usuarios; SELECT * FROM datos WHERE nombre LIKE '%
```

Código 3, en azul se refleja el dato a consultar y en rojo el código inyectado

```
consulta := SELECT * FROM usuarios WHERE nombre = 'Alicia';
DROP TABLE usuarios;
SELECT * FROM datos WHERE nombre LIKE '%';
```

Código 4, consulta resultante con la inyección de código SQL

Existe también una variante de este tipo de ataques, denominado Blind SQL Injection, donde se aprovecha de la falta de tratamiento, en el código SQL original, de los casos en que las consultas no obtengan resultados. Los atacantes se aprovechan de estas situaciones automatizando las peticiones, y conformando diccionarios con los elementos que sí que existen en las bases de datos.

Los ataques del estilo SQL Injection solo se pueden prevenir desde la programación segura, tanto de los entornos de autenticación como de los formularios que se procesen en el servicio, **en el lado del servidor**; esto se consigue empleando herramientas otorgadas por los distintos lenguajes de programación, que evitan la formación de las consultas SQL con *strings*. Sin embargo, siempre se debe tener en cuenta que es posible dificultar la obtención de los datos por parte de terceros, esto es posible almacenando los datos de los usuarios protegidos con mecanismos de cifrado seguros.

10.6. Ataque MiTM

El tipo de ataque conocido como Man in The Middle (MiTM), traducido al Castellano como *hombre en el medio*, **consiste en la intervención de las comunicaciones entre emisor y receptor por parte de un tercero**. Este ataque tiene la característica de que, **ninguno de los participantes de la comunicación, tiene constancia de que un tercero está escuchando la comunicación**.

En los casos en que no hay ningún tipo de seguridad en el canal de comunicación, basta con que un tercero escuche en el mismo para tener constancia de todo lo que intercambien emisor y receptor. Por este motivo se desarrollaron protocolos del tipo SSL, para establecer mecanismos de cifrado en los canales de comunicación, evitando así que un tercero comprenda el contenido de los mensajes enviados por el canal de comunicación.

El principal problema que conllevan los protocolos que aseguran los canales de comunicación, como por ejemplo SSL o TLS, viene a ser que se requiere de una fase de establecimiento de claves entre emisor y receptor, el llamado *handshake*. Los protocolos de establecimiento o *handshake* son aprovechados por este tipo de ataque, el *MiTM attack*. El atacante se encuentra a la espera de que se produzca un protocolo *handshake*, de forma que se le permita interponerse entre emisor y receptor, obteniendo así todos los parámetros de cifrado que negocien entre sí, y pudiendo escuchar perfectamente la comunicación entre emisor y receptor.

Para dificultar este tipo de ataques en los *handshake* se comenzaron a emplear los certificados de confianza, es decir, infraestructuras PKI. De esta forma le es posible a un cliente que intente acceder a un servicio, conocer que la entidad o servidor es quien dice ser.

Un atacante puede, aun así, falsificar un certificado o una CA de confianza, bien sea aprovechando fallos en el proveedor del servicio al que se conectarán los usuarios víctima, o bien aprovechando la falta de concienciación de los usuarios. Por poner un ejemplo de este último caso, la falta de concienciación de los usuarios, se tratará el caso en que un atacante realiza un MiTM en una red en que un usuario intenta acceder a un servicio de pago online.

Antes de proceder con el ejemplo, se debe tener en cuenta que los navegadores actuales poseen de serie los certificados y CAs de confianza, pues las empresas y proveedores que quieran hacer confiable su servicio deben pagar previamente por poseer certificados firmados por estas CAs consideradas “de confianza”.

- En primera instancia, el usuario navega por Internet de forma insegura, mediante el uso del protocolo HTTP, es decir, cualquier persona de su red podría escuchar la comunicación entre él y el servidor.

- El usuario, tras estar navegando, accede a una plataforma de pago que, por supuesto, exige una comunicación segura mediante TLS/SSL; procediendo con el establecimiento comenzará el *handshake*.

- El navegador del usuario, a partir de ahora denominado *cliente*, pedirá al servidor su certificado de confianza (tal y como se detalla en el apartado *9.1. Comunicaciones seguras: Protocolos SSL y TLS*).

- En este momento, un atacante a la espera de que se produzca el *handshake*, intercepta esa petición pasando a realizar el MiTM.

    - Replicará la petición del cliente hacia el servidor, suplantando al cliente de cara al servidor, pues el servidor no suele tener constancia de qué usuarios son los que intentan acceder a sus servicios.

    - Enviará un certificado falso al usuario, intentando suplantar al servidor de forma que el usuario proceda con el *handshake*; pudiendo así descifrar todos los datos que el cliente envíe al servidor, y cifrando los datos de nuevo hacia el servidor, creando así una ilusión de canal seguro de cara a las dos partes de la comunicación.

- Cuando el cliente recibe el certificado del atacante, detecta que no es un certificado confiable, por lo que le muestra al usuario una advertencia como la mostrada en la *imagen 20*. Si el usuario acepta la excepción, la comunicación proseguirá tal y como se ha detallado en el punto anterior, es decir, el atacante logrará la ilusión de canal seguro entre los dos participantes de la comunicación. Si el usuario rechaza la excepción, la comunicación no se establecerá, luego el atacante fracasará en su intento de MiTM.

Imagen 20, advertencia de posible MiTM

Cubriendo el caso que concierne a este documento, el sistema de autenticación segura en el CeSViMa, **hay una forma de evitar un ataque MiTM**. El principal problema que posee la detección de un MiTM es que, un servidor de un proveedor de servicios de Internet no tiene medios para conocer que el usuario no sea quien dice ser, por lo que un atacante tan solo tiene que engañar al usuario. Sin embargo, **dado que en este caso se trata de un servicio centralizado y reducido, donde los usuarios pueden poseer certificados almacenados por el servidor, los cuales deberán ser también firmados por una CA de confianza, el servidor podrá detectar y dificultar también los posibles ataques de MiTM**.

Otra posible medida anti-MiTM, sería el empleo de un sistema OTP, definido en el apartado *4.Informe de vigilancia sobre soluciones propuestas en el mercado actual*, ya que la autenticación de ambas partes en estos sistemas se basa en el previo conocimiento de una clave, que solo cliente y servidor conocen en cada momento por ser aleatoria y temporal.

11. Bibliografía

[1] Arkills, Brian. *LDAP Directories Explained: An Introduction and Analysis.* 2003.

[2] Migeon, Jean-Yves. *The MIT Kerberos Administrator's How-to Guide: Protocol, Installation and Single Sing On*. 2008.

[3] Rankl, Wolfgang; Effing, Wolfgang. *Smart Card Handbook (Third Edition)*. 2002.

[4] K. Jain, Anil; A. Ross, Arun; Nandakumar, Karthik. *Introduction to Biometrics*. 2011.

[5] Paar, Christof; Pelzl, Jan. *Understanding Cryptography: A Textbook for Students and Practitioners.* 2010.

[6] Adams, Carlisle; Lloyd, Steve. *Understanding PKI: Concepts, Standards, and Deployments Considerations (Second Edition)*. 2002.

[7] Enlace a definición del protocolo SSL

- http://www.cryptoheaven.com/Security/Presentation/SSL-protocol.htm

[8] Enlaces referentes a los problemas relacionados con el uso de passwords

- http://www.forbes.com/sites/tomkemp/2011/07/25/the-problems-with-passwords/
- http://www.wikidsystems.com/learn-more/Problem/passwords
- http://www.popularmechanics.com/technology/how-to/computer-security/solving-the-password-problem-14993917

[9] Enlace al top 500, listado de los 500 supercomputadores más potentes en Noviembre del 2013.

- http://www.top500.org/lists/2013/11/

[10] Enlaces referentes a los estándares ISO 27001 e ISO 27002

- http://es.wikipedia.org/wiki/ISO/IEC_27001
- http://www.iso.org/iso/catalogue_detail?csnumber=42103
- http://es.wikipedia.org/wiki/ISO/IEC_27002

- http://www.iso.org/iso/home/store/catalogue_ics/catalogue_detail_ics.htm?csnumber=39612

[11] Artículos publicados por Javier Hernández Martínez, abogado (**www.proteccionlegal.com**) con email **despacho@proteccionlegal.com**

- http://www.proteccionlegal.com/proteccion-de-datos/casos/145-proteccion-de-datos-sancion-por-almacenaje-indebido-de-contrasenas-y-aportacion-a-juicio.html
- http://www.proteccionlegal.com/proteccion-de-datos/casos/162-proteccion-de-datos-sancion-por-fax-sin-cifrar.html

[12] Caso PSN

- http://tecnologia.elpais.com/tecnologia/2011/04/26/actualidad/1303808469_850215.html
- http://www.fayerwayer.com/2011/04/espana-facua-exige-investigacion-contra-sony-tras-hackeo-de-psn/

[13] Caso Heartland Payment Systems

- http://en.wikipedia.org/wiki/Heartland_Payment_Systems
- http://www.elmundo.es/elmundo/2009/08/17/navegante/1250535175.html
- http://www.fayerwayer.com/2012/03/hackean-sistema-de-procesamiento-de-pagos-de-tarjetas-de-credito-de-visa-y-mastercard/

[14] Estadísticas de ataques informáticos en 2013

- http://seguinfo.wordpress.com/category/estadisticas/
- http://www.mcafee.com/us/resources/reports/rp-needle-in-a-datastack.pdf
- http://media.kaspersky.com/en/business-security/Kaspersky_Global_IT_Security_Risks_Survey_report_Eng_final.pdf
- http://www.networkworld.com/community/blog/fbiic3-impersonation-intimidation-and-scams-yep-that%E2%80%99s-internet-alright

[15] RFC referente al certificado X.509

- http://tools.ietf.org/html/rfc3280

[16] Enlace a explicación de la notación empleada en el estándar X.509

- http://en.wikipedia.org/wiki/Abstract_Syntax_Notation_One

[17] Enlace a la web del proyecto Nymi, basado en autenticación biométrica

- http://www.getnymi.com/

[18] Enlace a la web de la FNMT

- https://www.cert.fnmt.es

[19] Enlace a la referencia al bug 608 de OpenSSH

- https://bugzilla.mindrot.org/show_bug.cgi?id=608

[20] Enlace a la wiki de PuttyCard

- https://www.opensc-project.org/opensc/wiki/PuTTYcard

[21] Enlace a la web del proyecto Google Authenticator

- http://code.google.com/p/google-authenticator/

[22] Enlace de descarga del código de Google Authenticator

- https://code.google.com/p/google-authenticator/source/checkout

[23] Enlace a la web de Latch

- https://latch.elevenpaths.com/#

[24] Enlace a la publicación del NIST en el año 2011

- http://csrc.nist.gov/publications/nistpubs/800-131A/sp800-131A.pdf

[25] Enlace a la información sobre el supercomputador Magerit, proporcionada por la UPM

- http://www.upm.es/institucional/UPM/Centros/CampusMontegancedo/CESVIMA

[26] Enlace a la información sobre ataque sobre RSA en el 2011

- http://www.wired.com/threatlevel/2011/10/two-hacker-groups-breached-rsa/

[27] Enlace a Wikipedia donde se explica el algoritmo de generación de números aleatorios de los dispositivos de RSA, además de la puerta trasera que posee la NSA

- http://en.wikipedia.org/wiki/Dual_EC_DRBG